



TESIS - KI142502

**ANALISA SIMILARITAS MODEL PROSES BISNIS
MENGUNAKAN METODE HYBRID PROBABILISTIC
LATENT SEMANTIC ANALYSIS (PLSA) DAN
WEIGHTED DIRECTED ACYCLIC GRAPH (WDAG)**

Indra Gita Anugrah
NRP. 5114201006

DOSEN PEMBIMBING

Prof. Drs.Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.

PROGRAM MAGISTER

BIDANG KEAHLIAN MANAJEMEN INFORMASI

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2016

(halaman ini sengaja dikosongkan)



THESIS - KI142502

**BUSINESS PROCESS MODEL SIMILARITY ANALYSIS
USING HYBRID PROBABILISTIC LATENT SEMANTIC
ANALYSIS (PLSA) AND WEIGHTED DIRECTED
ACYCLIC GRAPH (WDAG) METHODS**

Indra Gita Anugrah
NRP. 5114201006

SUPERVISOR

Prof. Drs.Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.

MAGISTER PROGRAM

INFORMATION MANAGEMENT

DEPARTMENT OF INFORMATICS ENGINEERING

FACULTY OF INFORMATION TECHNOLOGY

INSTITUT TEKNOLOGI SEPULUH NOPEMBER SURABAYA

2016

(halaman ini sengaja dikosongkan)

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Komputer (M.Kom.)
di
Institut Teknologi Sepuluh Nopember Surabaya

oleh:
INDRA GITA ANUGRAH
Nrp. 5114201006

Dengan judul :
ANALISA SIMILIARITAS MODEL PROSES BISNIS MENGGUNAKAN METODE
HYBRID PROBABILISTIC LATENT SEMANTIC ANALYSIS (PLSA) DAN
WEIGHTED DIRECTED ACYCLIC GRAPH (WDAG)

Tanggal Ujian : 28-6-2016
Periode Wisuda : 2016 Genap

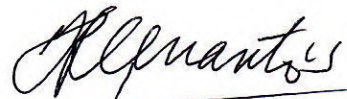
Disetujui oleh:

Prof.Ir.Drs.Ec. Riyanarto Sarno, M.Sc, Ph.D
NIP. 195908031986011001

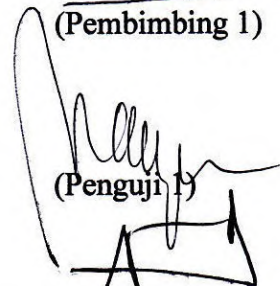
Dr. Ir. Raden Venantius Hari Ginardi, M.Sc
NIP. 196505181992031003

Dwi Sunaryono, S.Kom, M.Kom
NIP. 197205281997021001

Abdul Munif, S.Kom, M.Sc
NIP. 198608232015041004



(Pembimbing 1)



(Penguji 1)

(Penguji 2)



(Penguji 3)



Direktur Program Pasca Sarjana,

Prof.Ir.Djauhar Manfaat, M.Sc., Ph.D.
NIP. 196012021987011001

(halaman ini sengaja dikosongkan)

Analisa Similaritas Model Proses Bisnis Menggunakan Metode Hybrid Probabilistic Latent Semantic Analysis (PLSA) Dan Weighted Directed Acyclic Graph (WDAG).

Nama Mahasiswa : Indra Gita Anugrah

NRP : 5114 201 006

Pembimbing : Prof. Drs.Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.

Abstrak

Kebutuhan perusahaan atau organisasi dalam melakukan analisis model proses bisnis diharapkan agar perusahaan maupun organisasi dapat memahami proses yang berjalan serta dapat digunakan sebagai alat untuk membantu perusahaan dalam menghadapi perubahan dan perkembangan sehingga dapat mempermudah pengambilan kebijakan yang dibutuhkan dalam menghadapi perubahan dengan cepat, misalnya pemanfaatan proses yang telah berjalan untuk digunakan dalam proses lain (*reusable*). Tahap awal dalam penerapan *reusable* proses adalah mengetahui sub proses yang menyusun proses keseluruhan, tahap berikutnya melakukan analisis terhadap sub-proses dan menghitung nilai similaritas diantara sub-proses yang dibandingkan.

Pada penelitian ini akan dibahas metode dekomposisi untuk mengetahui sub-proses yang menyusun proses keseluruhan dari model proses bisnis menggunakan *Refined Process Structure Tree* (RPST) dan *Single Entry and Single Exit* (SESE), setelah itu dilakukan analisis menggunakan analisis gabungan dalam menentukan nilai similaritas diantara sub-proses. Metode analisis gabungan yang kami ajukan menggunakan *Weighted Directed Acyclic Graph* (WDAG) untuk analisa struktural dan *Probabilistic Latent Semantic Analysis* (PLSA) untuk analisa teekstual dengan tujuan mendapatkan hasil yang lebih baik dari pada menggunakan analisa struktural saja. Studi kasus dalam penelitian ini menggunakan proses bisnis Penerimaan Peserta Didik Baru (PPDB), dengan menggunakan metode diatas proses bisnis PPDB lebih mudah dianalisa sehingga membantu perusahaan atau organisasi dalam menghadapi perubahan secara cepat dan tepat.

Kata kunci: *Reusable Bisnis Proses, Dekomposisi, Refined Process Structured Tree (RPST), Probabilistic Latent Semantic Analysis (PLSA), Weighted Directed Acyclic Graph (WDAG).*

(halaman ini sengaja dikosongkan)

***Hybrid Similarity Analysis Method of Business Process Model Using
Probabilistic Latent Semantic Analysis (PLSA) and Weighted Directed Acyclic
Graph (WDAG)***

Student Name : Indra Gita Anugrah
NRP : 5114 201 006
Supervisor : Prof. Drs.Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.

Abstract

Needs of the company or organization in analyzing business process models is expected that companies and organizations can understand the processes that are running and can be used as a tool to help companies in the face of change and development so as to facilitate the decision making required in the face of change quickly, for example, the use of process have gone on to be used in another process (*reusable*). The initial stage in the application process is knowing reusable sub-processes that make up the whole process, the next stage of the analysis of sub-processes and calculate the value of similarity between the sub-processes that are compared.

In this research will be discussed decomposition method to determine the sub-processes that make up the overall process of business process models using the *Refined Process Structure Tree* (RPST) and *Single Entry and Single Exit* (SESE), then analyzed using analysis combined in determining the value of similarity between sub process. The combined analysis method that we propose to use *Weighted Directed Acyclic Graph* (WDAG) for structural analysis and *Probabilistic Latent Semantic Analysis* (PLSA) for textual analysis in order to get better results from the structural analysis only. The case studies in this study using a business process Admission New Students (PPDB), using the above method PPDB business processes more easily analyzed and can help change quickly and accurately with the development

Keywords: *Reusable Business Process, Decomposition, Refined Process Structured Tree (RPST), Probabilistic Latent Semantic Analysis (PLSA), Weighted Directed Acyclic Graph (WDAG).*

(halaman ini sengaja dikosongkan)

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT yang telah melimpahkan rahmat dan hidayah-Nya sehingga buku tesis ini dapat diselesaikan dengan baik. Meski dalam menyelesaikan buku ini banyak ditemui kesulitan, namun berkat bantuan dan bimbingan berbagai pihak, akhirnya saya berhasil menyelesaikan buku ini. Untuk itu atas segala bantuan yang telah diberikan, saya mengucapkan terima kasih serta penghargaan yang sebesar-besarnya antara lain kepada:

1. Kedua Orang tuaku, Kedua Mertuaku, Istri dan Anak tercinta, serta seluruh keluarga besar atas dukungan dan doanya sehingga saya bisa mendapatkan kemudahan dalam menyelesaikan studi ini. Semoga Allah SWT senantiasa memberikan rahmat, kesehatan, dan keselamatan kepada kita semua.
2. Bapak Prof. Drs.Ec. Ir. Riyanarto S, M.Sc., Ph.D., selaku dosen pembimbing yang telah banyak meluangkan waktu dan pikiran sampai terselesaikannya tesis ini. Semoga Allah SWT senantiasa merahmati dan memberikan kesehatan Kepada Bapak dan Keluarga.
3. Bapak Dr. Ir Raden Venantius H. G. M.Sc, Bapak Dwi Sunaryono. S.Kom, M.Kom, dan Bapak Abdul Munif, S.Kom, M.Sc, selaku dosen penguji yang telah banyak membantu saya untuk menjadi lebih baik.
4. Seluruh dosen S2 Teknik Informatika ITS yang telah memberikan wawasan serta ilmu pengetahuan baru bagi saya selama menempuh masa studi pascasarjana.
5. Teman seperjuangan angkatan 2014 yang telah berbagi dan saling mendukung serta menyemangati dalam masa masa perkuliahan hingga masa penulisan tesis.

Saya menyadari bahwa dalam laporan tesis ini masih banyak kekurangannya, karena itu masukan, saran demi perbaikan dan penerapan tesis ini dimasa mendatang tetap saya harapkan. Semoga tesis ini dapat benar-benar bermanfaat bagi agama, masyarakat, bangsa dan negara.

Surabaya, Agustus 2016

Penulis

DAFTAR ISI

| | |
|--|------|
| LEMBAR PENGESAHAN | v |
| ABSTRAK | vii |
| ABSTRACT | ix |
| KATA PENGANTAR | xi |
| DAFTAR ISI | xiii |
| DAFTAR GAMBAR | xvii |
| DAFTAR TABEL | xix |
| BAB 1 PENDAHULUAN | 1 |
| 1.1 Latar Belakang | 1 |
| 1.2 Perumusan Masalah | 3 |
| 1.3 Tujuan | 4 |
| 1.4 Manfaat Penelitian | 4 |
| 1.5 Kontribusi Penelitian | 4 |
| 1.6 Batasan Masalah | 5 |
| BAB 2 KAJIAN PUSTAKA | 7 |
| 2.1 <i>Workflow Management System</i> (WMS) | 7 |
| 2.2 Dekomposisi menggunakan RPST dan SESE | 8 |
| 2.2.1 <i>Refined Process Structured Tree</i> (RPST) | 8 |
| 2.2.1.1 <i>Fragments</i> | 9 |
| 2.2.1.2 <i>Triconnected Component</i> | 9 |
| 2.2.2 <i>Single Entry and Single Exit</i> (SESE) | 10 |
| 2.3 Model Abstraksi dari <i>Behavioral Models</i> | 11 |
| 2.3.1 <i>Triconnected Abstraction</i> | 11 |
| 2.4 Analisa Similiaritas Proses Bisnis | 13 |
| 2.4.1 <i>Process Vector</i> | 13 |
| 2.4.1.1 Matrik <i>Business Process Control Flow Complexity</i> | 14 |
| 2.4.1.2 Kompleksitas Sebuah Proses | 14 |
| 2.4.1.3 Pengukuran Kompleksitas Proses Dalam Proses Bisnis | 14 |
| 2.5 Model Ruang Vektor | 15 |
| 2.6 Pemobobotan Kata (Term) | 16 |

| | | |
|----------------------------------|--|----|
| 2.7 | <i>Latent Semantic Analysis (LSA)</i> | 17 |
| 2.7.1 | <i>Singular Value Decomposition</i> | 18 |
| 2.7.2 | <i>Eigenvalue dan Eigenvector</i> | 18 |
| 2.7.3 | <i>Reduksi Dimensi</i> | 18 |
| 2.8 | <i>Probabilistic Latent Semantic Analysis (PLSA)</i> | 19 |
| 2.9 | <i>Metode Klasifikasi Naïve Bayes</i> | 23 |
| 2.10 | <i>Weighted Directed Acyclic Graph (WDAG)</i> | 24 |
| 2.10.1 | <i>Teori Graph</i> | 24 |
| 2.10.2 | <i>Perhitungan Similiaritas Weighted Directed Acyclic Graph (WDAG)</i> | 25 |
| 2.11 | <i>Penelitian Terkait</i> | 28 |
| BAB 3 METODE PENELITIAN | | 29 |
| 3.1 | <i>Studi Literatur</i> | 31 |
| 3.2 | <i>Desain Sistem</i> | 32 |
| 3.2.1 | <i>Pengumpulan Dataset</i> | 32 |
| 3.2.2 | <i>Pra-Proses</i> | 32 |
| 3.2.2.1 | <i>Proses Bisnis Menggunakan Pemodelan Workflow Net</i> | 34 |
| 3.2.2.2 | <i>Dekomposisi Menggunakan RPST dan SESE</i> | 34 |
| 3.2.3 | <i>Analisa Secara Struktural</i> | 34 |
| 3.2.3.1 | <i>Process Vector</i> | 37 |
| 3.2.3.2 | <i>Triconnected Abstraction</i> | 38 |
| 3.2.4 | <i>Analisa Secara Tekstual (Semantik Sintaksis)</i> | 40 |
| 3.2.4.1 | <i>Probabilistic Latent Semantic Analysis (PLSA)</i> | 41 |
| 3.2.4.1.1 | <i>Metode Naive Bayes</i> | 42 |
| 3.2.4.1.2 | <i>Metode Expection Maximization</i> | 43 |
| 3.2.4.1.2.1 | <i>Expection Step</i> | 43 |
| 3.2.4.1.2.2 | <i>Maximization Step</i> | 45 |
| 3.2.5 | <i>Perhitungan Similaritas WDAG</i> | 51 |
| 3.3 | <i>Pengujian, Hasil dan Analisa</i> | 51 |
| 3.4 | <i>Pembuatan Perangkat Lunak</i> | 52 |
| BAB 4 HASIL DAN PEMBAHASAN | | 53 |

| | |
|---|----|
| 4.1 Hasil Penelitian | 53 |
| 4.1.1 Komposisi Dalam Proses Bisnis PPDB | 54 |
| 4.1.2 Hasil Dekomposisi Menggunakan RPST Dan SESE..... | 54 |
| 4.1.3 Hasil Pembobotan <i>Edge</i> WDAG Menggunakan WCDG..... | 57 |
| 4.1.4 Hasil Representasi WDAG | 58 |
| 4.1.5 Hasil Pemodelan Topik Menggunakan PLSA | 60 |
| 4.2 Pembahasan Hasil Pengujian | 62 |
| 4.2.1 Perbandingan Perhitungan WDAG Similarity | 62 |
| 4.2.2 Perbandingan Metode <i>Tekstual Similarity</i> | 63 |
| 4.2.3 Pengaruh Process Vector Terhadap Perhitungan WDAG..... | 65 |
| 4.2.4 Pengaruh <i>Tekstual Similarity</i> Terhadap Perhitungan WDAG | 65 |
| BAB 5 PENUTUP | 67 |
| 5.1 Kesimpulan | 67 |
| 5.2 Saran..... | 68 |
| DAFTAR PUSTAKA | 69 |
| LAMPIRAN..... | 71 |
| DAFTAR RIWAYAT HIDUP..... | 77 |

(halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

| | | |
|-------------|---|----|
| Gambar 2.1 | Contoh <i>Petri Net</i> Dan <i>Workflow Net</i> | 8 |
| Gambar 2.2 | (a) <i>Two Terminal Graph</i> , dan (b) Hasil RPST dari (a) | 8 |
| Gambar 2.3 | Contoh <i>Triconnected Component</i> (a) <i>Polygon</i> , (b) <i>Rigid</i> , (c) <i>Bond Component</i> | 10 |
| Gambar 2.4 | <i>Fragment Single Entry and Single Exit</i> (SESE)..... | 11 |
| Gambar 2.5 | Model Abstraksi dari <i>Trivial</i> | 12 |
| Gambar 2.6 | Model Abstraksi dari <i>Trivial</i> | 12 |
| Gambar 2.7 | Model Abstraksi dari <i>Bond</i> | 12 |
| Gambar 2.8 | Model Abstraksi dari <i>Rigid</i> | 13 |
| Gambar 2.9 | Representasi Vektor Dokumen dan <i>Query</i> | 16 |
| Gambar 2.10 | Sebuah Matrik A | 18 |
| Gambar 2.11 | Nilai <i>Eigenvalue</i> dan <i>Eigenvector</i> dari Matrik A..... | 19 |
| Gambar 2.12 | Reduksi Dimensi dari Sebuah Matriks..... | 19 |
| Gambar 2.13 | Hubungan antar Dokumen, Topik, dan Kata..... | 20 |
| Gambar 2.14 | Matrik $P(d z)$ | 21 |
| Gambar 2.15 | Contoh dari <i>Graph</i> | 24 |
| Gambar 2.16 | Penelitian Terkait | 28 |
| Gambar 3.1 | Tahapan Metodologi Penelitian | 31 |
| Gambar 3.2 | Desain Sistem..... | 33 |
| Gambar 3.3 | Alur Pra-Proses | 34 |
| Gambar 3.4 | Alur Analisa <i>Struktural</i> | 34 |
| Gambar 3.5 | Model <i>Pre Registration</i> 1 dan 3 | 35 |
| Gambar 3.6 | Hasil Dekomposisi Model <i>Fragment Pre Registration</i> | 36 |
| Gambar 3.7 | RPST <i>Pre Registration</i> | 37 |
| Gambar 3.8 | Model <i>Fragment</i> | 37 |
| Gambar 3.9 | <i>Triconnected Abstraction Fragment</i> | 39 |
| Gambar 3.10 | Alur Analisa <i>Tekstual</i> PLSA | 41 |
| Gambar 3.11 | Representasi WDAG | 50 |

| | |
|---|----|
| Gambar 4.1. Hasil Dekomposisi Model <i>Pre Registration</i> | 54 |
| Gambar 4.2. <i>Triconnected Abstraction Fragment Pre Registration</i> | 59 |

DAFTAR TABEL

| | | |
|-------------|--|----|
| Tabel 2.1 | Matrik <i>Control Flow Complexity</i> Proses Bisnis | 15 |
| Tabel 3.1 | <i>Prosecess Vector Model Fragment</i> | 37 |
| Tabel 3.2 | Label <i>String Node Model Pre Registration 1</i> | 40 |
| Tabel 3.3 | Label <i>String Node Model Pre Registration 2</i> | 40 |
| Tabel 3.4 | Perhitungan <i>Cosine Measure</i> Label <i>String Node B</i> | 40 |
| Tabel 3.5 | Contoh Prediksi Kategori..... | 42 |
| Tabel 3.6 | Matriks Term Dokumen..... | 44 |
| Tabel 3.7 | Nilai Probabilitas | 44 |
| Tabel 3.8 | Nilai Probabilitas Setelah <i>EM Step</i> | 48 |
| Tabel 3.9 | <i>Confusion Matrix</i> | 51 |
| Tabel 4.1.a | Sub proses <i>Fragment</i> | 55 |
| Tabel 4.1.b | Sub proses <i>Fragment</i> Lanjutan | 56 |
| Tabel 4.2. | Jumlah <i>Fragment</i> | 57 |
| Tabel 4.3. | Proses pembobotan “FRAGMENT1_PR” | 58 |
| Tabel 4.4. | Probabilitas Awal <i>Term</i> Terhadap Topik..... | 60 |
| Tabel 4.5. | Probabilitas Awal Dokumen Terhadap Topik | 60 |
| Tabel 4.6. | Probabilitas <i>Term</i> Terhadap Topik - <i>EM Step</i> | 61 |
| Tabel 4.7. | Probabilitas Dokumen Terhadap Topik - <i>EM Step</i> | 61 |
| Tabel 4.8. | Contoh Hasil Uji Coba Perhitungan WDAG | 62 |
| Tabel 4.9. | Matiks Perbandingan Untuk Pengukuran Kinerja Sistem..... | 63 |
| Tabel 4.10. | Contoh Perbandingan Hasil <i>Tekstual Similarity</i> | 64 |

(halaman ini sengaja dikosongkan)

DAFTAR LAMPIRAN

| | | |
|---------------|---|----|
| Lampiran L1 | Model PPDB | 71 |
| Lampiran L1.1 | Contoh Model Pre Registration..... | 71 |
| Lampiran L1.2 | Contoh Variasi Model Pre Registration | 72 |
| Lampiran L1.3 | Contoh Model Registration | 73 |
| Lampiran L1.4 | Contoh Variasi Model Registration | 74 |
| Lampiran L1.5 | Contoh Model Student Collection..... | 75 |
| Lampiran L1.6 | Contoh Variasi Model Student Collection | 76 |

(halaman ini sengaja dikosongkan)

BAB I

PENDAHULUAN

Pada bab ini akan dijelaskan mengenai beberapa hal dasar dalam pembuatan proposal penelitian yang meliputi latar belakang, perumusan masalah, tujuan, manfaat, kontribusi penelitian, dan batasan masalah.

1.1 Latar Belakang

Penerapan Teknologi Informasi dalam sebuah perusahaan merupakan kebutuhan yang utama, terutama dewasa ini dimana perusahaan mendapatkan tantangan, khususnya yang datang dari luar misalnya permintaan pelanggan dan persaingan pasar, sehingga mendorong perusahaan untuk melakukan inovasi untuk mencapai kepuasan pelanggan, apabila hal ini tidak dilakukan maka sebuah perusahaan akan kalah bersaing dengan perusahaan kompetitor. Untuk melakukan inovasi dibutuhkan upaya untuk menjalankan proses bisnis secara optimal, hal ini dimungkinkan dilakukan dalam Bisnis Proses Manajemen (BPM). Proses bisnis adalah sebuah instrumen untuk mengatur segala aktifitas serta meningkatkan pemahaman dari keterhubungan diantara bagian atau proses (Weske, 2007), sedangkan BPM adalah sebuah pendekatan yang menyeluruh untuk meningkatkan efektivitas dan efisiensi dalam bisnis sebuah perusahaan. BPM bertujuan agar proses berjalan optimal, fleksibel, *reusable* dan terintegrasi dengan teknologi Informasi.

Tantangan perusahaan dalam penerapan *Enterprise Resource Planning* (ERP), dimana ERP harus dapat dikonfigurasi ulang secara cepat dan fleksibel (Sarno, Djeni, Mukhlas, & Sunaryono, 2015), hal ini dimungkinkan dengan menggunakan proses yang sudah ada untuk diterapkan dalam proses lain yang berbeda (*reusable*). Penggunaan *reusable* proses bisnis dapat dilakukan dengan menemukan proses yang mempunyai kemiripan (*similarity*) yang sama, untuk menemukan proses yang mempunyai kemiripan, dapat dilakukan dengan menganalisis setiap proses yang ada, permasalahan yang sering terjadi dalam analisis proses bisnis adalah permasalahan kompleksitas dari proses yang berjalan dalam perusahaan, proses bisnis yang berjalan dalam suatu perusahaan terdiri dari

kumpulan proses yang menggambarkan hubungan antara bagian, sehingga sangat sulit dianalisis karena memiliki tingkat kompleksitas yang tinggi, karena itu diperlukan sebuah metode untuk memecah proses bisnis keseluruhan menjadi sub-proses dengan tujuan lebih mudah dalam melakukan analisis untuk menemukan proses yang mempunyai kemiripan diantara model proses yang dibandingkan.

Pada penelitian ini untuk memecah atau menguraikan proses bisnis keseluruhan menjadi sub-proses, kami menggunakan metode yang dikemukakan oleh (Vanhatalo, Völzer, & Koehler, 2009), yaitu *Refined Process Structured Tree* (RPST) dimana *graph* yang mempunyai keterhubungan lemah dapat diuraikan kedalam sekumpulan *graph* yang dinamakan *Triconnected Component* yang memiliki satu masukan dan satu keluaran *Single Entry and Single Exit* (SESE), dalam penelitian (Munoz-Gama, Carmona, & Aalst, 2014) menggunakan SESE yang digunakan sebagai pengecekan kesesuaian (*Conformance Checking*), hasil dari penguraian dari proses bisnis keseluruhan adalah sub-proses atau *fragment*, sub-proses inilah yang akan kita analisis untuk menentukan tingkat kemiripan sehingga kita dapat menggunakan sub-proses untuk digunakan pada sub-proses lain yang memiliki nilai kemiripan yang tinggi, sebagaimana tujuan dari penelitian ini tentang membangun *reusable* proses bisnis.

Tahap selanjutnya adalah bagaimana cara melakukan analisis kemiripan dari model proses, banyak penelitian dilakukan dalam pengukuran kemiripan model proses, salah satunya mengacu pada penelitian yang dilakukan oleh (Dijkman, Dumas, Dongen, Reina, & Mendling, 2011), untuk mengukur tingkat kemiripan dari model proses dapat dilakukan menggunakan tiga pendekatan yaitu: pendekatan analisis kemiripan secara sintaksis, semantik dan kontekstual, selain pendekatan diatas, untuk melakukan analisis kemiripan dapat dilakukan menggunakan pendekatan struktural. Analisis pendekatan struktural biasanya dilakukan menggunakan pendekatan struktur *graph* sebagaimana penelitian yang dilakukan oleh (Dijkman, Dumas, & García-Bañuelos, 2009) dimana pendekatan analisis kemiripan secara struktural berdasarkan topologi model proses yang dilihat sebagai *graph*. Pendekatan analisis secara struktural lainnya dilakukan menggunakan *Process Structure Tree* (PST) sebagaimana penelitian yang dilakukan oleh (Ling, Zhang, & Feng, 2014).

Beberapa penelitian tentang pendekatan analisis kemiripan diantara model proses menggunakan analisis secara struktural memiliki kelemahan, di mana model proses yang dibandingkan mungkin mempunyai nilai kemiripan yang tinggi secara struktural namun secara kenyataan belum tentu demikian, karena itu pada penelitian ini kami mengajukan sebuah metode penggabungan metode analisis kemiripan secara struktural dengan metode analisis secara *tekstual* (semantik sintaksis). Metode analisis struktural yang kami gunakan adalah metode yang diajukan oleh (Jung, Bae, & Liu, 2009), di mana analisis struktural dilakukan menggunakan pemodelan proses berbasis *graph* di mana model proses bisnis diubah kedalam *Process Vector*, sedangkan analisis secara *tekstual* (semantik sintaks) kami menggunakan metode *Probabilistic Latent Semantic Analysis* (PLSA). PLSA merupakan sebuah metode berbasis pemodelan topik yang bertujuan merepresentasikan dokumen, menurut (Hofmann, 1999), ide permulaan dari PLSA adalah sebuah model aspek yang direpresentasikan menggunakan metode statistik setelah dilakukan analisis kemiripan secara struktural dan *tekstual* dibutuhkan sebuah metode untuk merepresentasikan dan melakukan analisis kemiripan gabungan kedalam *Weighted Direct Acyclic Graph* (WDAG), serta dilakukan perhitungan tingkat kemiripan menggunakan metode yang dikemukakan oleh (Jin, 2006).

Tujuan dari metode gabungan ini adalah menghasilkan nilai analisis kemiripan yang lebih baik dibandingkan hanya menggunakan pendekatan secara struktural saja.

1.2 Perumusan Masalah

Hipotesis dalam penelitian ini adalah dengan menggabungkan metode analisis kemiripan secara struktural dan secara *tekstual* dapat menambah akurasi terhadap tingkat kemiripan dari model proses, adapun metode analisis struktural dilakukan dengan memetakan *Process Vector*, dimana *Process Vector* terdiri dari *Activity Vector* dan *Transition Vector*, sedangkan analisis kemiripan secara *tekstual* dilakukan menggunakan PLSA untuk menemukan makna/arti dari setiap label dari *activity* dan *transition* dari model proses, setelah analisis kemiripan secara struktural dan *tekstual* dilakukan, tahap berikutnya merepresentasikan

kedalam *Weighted Direct Acyclic Graph* (WDAG), berdasarkan hipotesis diatas, maka rumusan masalah dalam penelitian ini dapat dijabarkan menjadi beberapa poin berikut ini:

1. Bagaiman merepresentasikan model proses kedalam *Process Vector*?
2. Bagaimana melakukan analisis secara *tekstual* terhadap label dari setiap bagian dari model proses dengan menggunakan metode PLSA?
3. Bagaimana merepresentasikan analisis kemiripan gabungan ke dalam *Weighted Direct Acyclic Graph* (WDAG)?
4. Bagaimana melakukan perhitungan kemiripan dari *Weighted Direct Acyclic Graph* (WDAG)?
5. Bagaimana cara melakukan evaluasi dari metode yang diajukan?

1.3 Tujuan

Tujuan dari tesis ini adalah melakukan analisis kemiripan proses bisnis menggunakan metode gabungan analisis kemiripan secara struktural dengan *tekstual*, dengan harapan metode gabungan ini dapat menghasilkan nilai analisis kemiripan yang lebih baik dibandingkan hanya menggunakan pendekatan secara struktural saja.

1.4 Manfaat

Penelitian ini diharapkan dapat menghasilkan metode analisis perhitungan kemiripan dari model proses yang lebih baik, dengan mengetahui kemiripan dari model proses akan bermanfaat bagi suatu perusahaan atau organisasi dalam penerapan *reusable* model proses.

1.5 Kontribusi Penelitian

Kontribusi dalam penelitian ini yang terkait dengan analisis kemiripan dari model proses adalah membangun model baru yaitu model analisis kemiripan gabungan dengan menggabungkan analisis kemiripan struktural dan *tekstual*, selain membangun model analisis kemiripan gabungan, kontribusi lainnya adalah memperbaiki perhitungan kemiripan *Weighted Direct Acyclic Graph* (WDAG).

1.6 Batasan Masalah

Batasan dalam penelitian ini adalah sebagai berikut:

1. Penelitian ini menekankan pada penggabungan analisis kemiripan struktural dan *tekstual* kemudian direpresentasikan dan dilakukan perhitungan nilai kemiripan gabungan menggunakan metode *Weighted Direct Acyclic Graph (WDAG) Similarity*.
2. Studi kasus dari penelitian ini menggunakan proses bisnis Penerimaan Peserta Didik Baru online (PPDB).
3. Dataset kamus kata (*Bag Of Word*) yang digunakan sebagai pemodelan topik dalam *Probalistic Latent Semantic Analysis (PLSA)* menggunakan label *string* dari *node* dan *edge* proses bisnis PPDB.
4. Sistem yang dibangun menggunakan bahasa pemrograman Java.

(halaman ini sengaja dikosongkan)

BAB II

KAJIAN PUSTAKA DAN DASAR TEORI

Pada bab 2 ini dijelaskan konsep dasar tentang teori dan kajian pustaka yang digunakan sebagai landasan dalam melakukan penelitian.

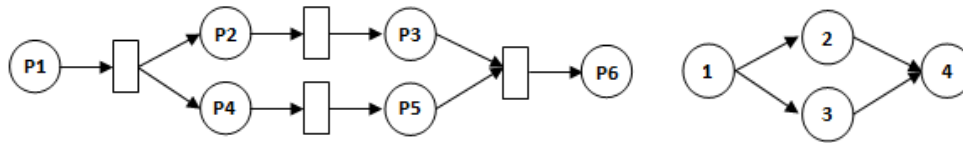
2.1. *Workflow Management System (WMS)*

Perusahaan atau organisasi umumnya terdiri dari sekumpulan proses antara bagian dengan tujuan untuk mencapai kepentingan sebuah perusahaan atau organisasi, dalam penerapan sistem informasi, semakin besar sebuah perusahaan maka semakin kompleks proses di dalamnya sehingga dibutuhkan sebuah manajemen proses untuk menganalisis kebutuhan yang diperlukan oleh perusahaan yang dinamakan *Workflow Management System (WMS)*, tujuan utama dari penerapan WMS adalah untuk mendukung rincian pelaksanaan dan kontrol dari proses (Aalst, 1998), dalam pelaksanaannya dibutuhkan pemodelan untuk menggambarkan proses yang ada di dalamnya. *Petri Net* merupakan salah satu pemodelan yang dapat digunakan untuk memodelkan proses.

- *Petri Net* terdiri dari (P, T, F) :
 - **P** adalah himpunan dari *Places* yang dinotasikan dengan simbol lingkaran.
 - **T** adalah himpunan dari *Transition* yang dinotasikan dengan simbol bujur sangkar.
 - **F** adalah himpunan dari busur yang dinotasikan dengan simbol panah.
- *Workflow Net* terdiri dari (P, T, A) :

Workflow Net adalah jenis tertentu dari *Petri Net*, di mana *Net* memiliki sebuah *Places* 'start' dan sebuah *Places* 'end', dan semua *nodes* lain yang berada dalam jalur di antaranya.

 - **A** merupakan hubungan alur/*flow relation* di antara *Places* (*P*) dan *Transition* (*T*).



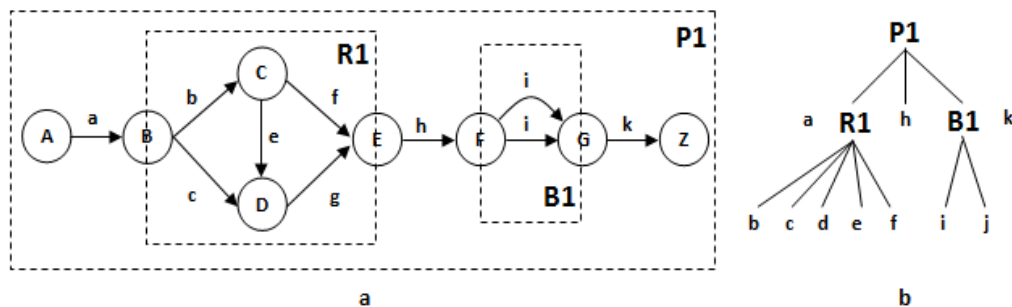
Gambar 2.1 Contoh *Petri Net* dan *Workflow Net*

2.2 Dekomposisi Menggunakan RPST dan SESE

Permasalahan yang muncul dalam pemodelan bisnis proses dalam sebuah perusahaan atau organisasi, salah satunya adalah tingkat kompleksitas dari bisnis proses itu sendiri, di mana dalam bisnis proses suatu perusahaan atau organisasi menggambarkan hubungan proses di antara bagian yang ada sehingga sangat sulit jika menganalisis bisnis proses keseluruhan secara langsung sehingga dibutuhkan sebuah metode untuk menguraikan/memecah bisnis proses menjadi bagian yang lebih kecil yang dinamakan *fragment* model proses, semakin rendah nilai kompleksitas dari suatu proses maka akan semakin mudah dianalisis untuk dilakukan pemeliharaan dan dibandingkan dengan model keseluruhan dari proses bisnis.

2.2.1 *Refined Process Structured Tree* (RPST)

Metode untuk menguraikan sebuah proses yang kompleks dapat dilakukan dengan menggunakan metode *Refined Process Structured Tree* (RPST), metode ini dikemukakan oleh (Vanhatalo, Völzer, & Koehler, 2009), di mana penguraian berdasarkan *Two Terminal Graph* (TTG) dan hasil dari penguraian menggunakan RPST merupakan sebuah kumpulan dari *fragment* objektif yang dinamakan *fragment Triconnected Component*, sebuah *fragment* dikatakan objektif jika tidak terdapat *fragment* yang tidak saling tumpang tindih.



Gambar 2.2 (a) *Two Terminal Graph*, dan (b) Hasil RPST dari (a)

2.2.1.1 Fragments

Workflow Graph dibangun berdasarkan dari *Two Terminal Graph* (TTG), TTG adalah sebuah graph berarah yang tidak berulang pada dirinya sehingga terdapat *Source Nodes* dan *Sink Nodes* yang tidak sama, jika *Source Nodes* adalah A dan *Sink Nodes* adalah Z maka $A \neq Z$ dan setiap *node* alfabet mengarah dari A ke Z.

Definisi 1 (*Boundary Nodes, Entry, Exit dan Fragment*):

1. *Boundary Nodes* adalah sebuah *Source Nodes* atau *Sink Nodes* dari *graph*.
2. Sebuah *Boundary Nodes* dikatakan *Entry* jika tidak ada *edges* yang mengarah (*incoming*) pada *Source Nodes* atau jika semua *edges* meninggalkan *Source Nodes* (*outgoing*).
3. Sebuah *Boundary Nodes* dikatakan *Exit* jika semua *edges* yang mengarah pada *Sink Nodes* atau jika tidak ada *edges* yang meninggalkan *Sink Nodes* (*outgoing*).
4. Sebuah sub-*graph* dikatakan sebuah *fragment* jika memiliki tepat dua *Boundary Nodes*, sebagai *Entry* dan *Exit*.

2.2.1.2 Triconnected Component

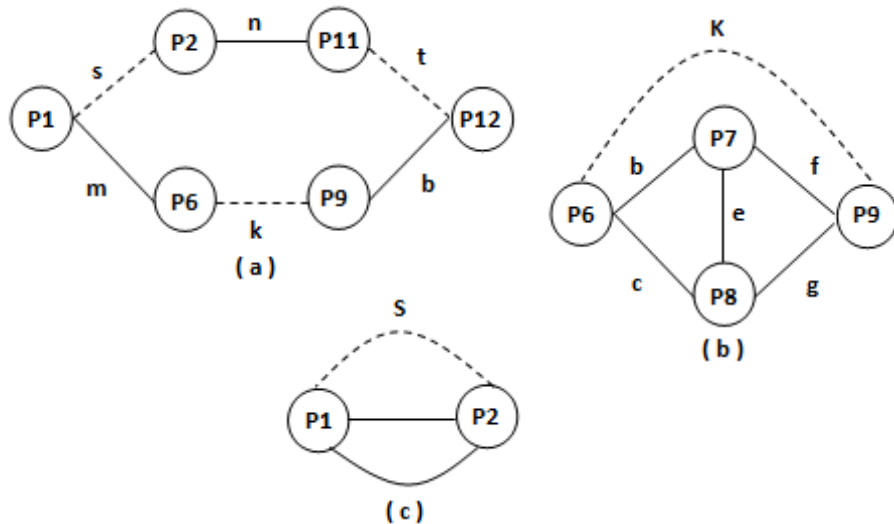
Sebuah *graph* yang tidak terhubung dapat diuraikan kedalam beberapa bagian yang terhubung secara unik kedalam dua buah sub-*graph* saling terhubung (*Biconnected Graph*), sebuah *fragment* dari TTG erat kaitanya dengan dengan *Triconnected Component* oleh karena itu diperlukan untuk menguraikan *Biconnected Graph* menjadi *Triconnected Components* karena keterhubungan ini sangat menentukan dalam perhitungan RPST (Polyvyanyy, 2012). Gambar 2.2 (a) merupakan TTG yang terdiri dari 2 buah sub-*graph*, dimana sub-*graph* satu memiliki *Boundary Source Nodes* B dan *Boundary Sink Nodes* E sedangkan sub-*graph* dua memiliki *Boundary Source Nodes* F dan *Sink Nodes* G, di mana antara *Boundary Sink Nodes* B dan *Boundary Source Nodes* F di hubungkan dengan *edges* hal ini dinamakan *Biconnected Graph*.

Triconnected Component merupakan hasil penguraian dari *graph* yang mempunyai keterhubungan yang lemah (*weakly biconnected graph*) dimana tidak

terdapat *edges* dari sekumpulan *nodes* sub-graph dua yang mengarah tepat ke sekumpulan *nodes* sub-graph satu dan terdapat *virtual edges* berupa garis putus antara *Boundary Source Nodes* B dan *Boundary Sink Nodes* G. *Triconnected Component* dari sebuah *graph* dapat berupa *Polygon*, *Rigid*, *Bond* dan *Trivial Component* (Polyvyanyy, 2012) dimana:

Definisi 2 (*Trivial*, *Bond*, *Polygon*, dan *Trivial Component*) :

1. *Trivial* adalah sebuah *fragments/sub-graph* yang hanya memiliki sebuah *single edges*.
2. *Bond* adalah *fragments/sub-graph* yang terdiri dari dua *nodes* dan *edges* sejumlah $k \geq 2$ di antaranya.
3. *Polygon* adalah *fragments/sub-graph* yang terdiri dari *nodes* dan *edges* sejumlah $k \geq 3$, sehingga semua *nodes* dan *edges* terkandung dalam siklus (lingkaran).
4. *Rigid* adalah *fragments/sub-graph* yang terdiri dari *Bond* dengan tiga *edges* atau sebuah *simple Triconnected Graph*, simple berarti bahwa tidak ada pasangan *nodes* yang di hubungkan oleh lebih dari satu *edges*

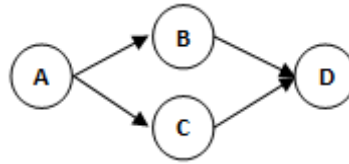


Gambar 2.3 Contoh *Triconnected Component* (a) *Polygon*, (b) *Rigid*, (c) *Bond Component*

2.2.2 *Single Entry and Single Exit* (SESE)

Single Entry and Single Exit (SESE) merupakan sebuah *Workflow Graph* dimana memiliki satu masukan dan satu keluaran (*Single Entry and Single Exit*) dalam penelitian yang dilakukan oleh (Munoz-Gama, Carmona, & Aalst, 2014)

digunakan sebagai pengecekan kesamaan/*Conformance Checking*. Pada Gambar 2.4 dijelaskan bahwa *Node A* merupakan *Node Single Entry* dan *Node D* merupakan *Node Single Exit*.



Gambar 2.4 *Fragment Single Entry and Single Exit*

2.3. Model Abstraksi dari *Behavioral Models*

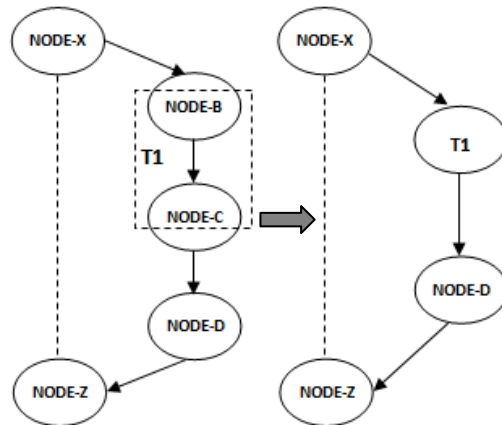
Behavioral Models dikembangkan dengan berbagai tujuan, salah satunya di gunakan sebagai acuan dalam melakukan proses desain ulang (*redesign*) dan sebagai instruksi yang tepat dalam eksekusi sebuah proses. *Behavioral Models* berfungsi untuk menggambarkan prosedur yang bekerja secara detail. Pada penelitian (Polyvyanyy, 2012) mengemukakan sebuah metode *Abstraction Model* dari *Triconnected Component* untuk menggambarkan *Behavioral Models* dengan menerapkan *Transformation Rule* yang dapat disesuaikan dengan kebutuhan.

Abstraction adalah sebuah pendekatan untuk mengurangi detail yang tidak diinginkan dengan mempertahankan informasi yang penting. Informasi yang penting adalah yang diperlukan oleh pihak tertentu dalam melakukan tugasnya. Tujuan dari *Abstraction* adalah memberi gambaran tentang proses mana yang dapat direduksi.

2.3.1 *Triconnected Abstraction*

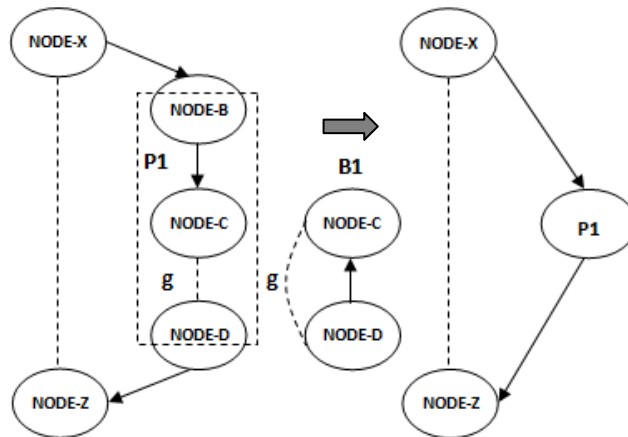
Ide dari *Triconnected Abstraction* adalah pertukaran sebuah *fragment Single Entry and Single Exit* dengan sebuah *Abstraction Model* setiap proses pertukaran tersebut disebut langkah abstraksi. Langkah abstraksi tersebut dipicu oleh *task node* yang mempunyai paling banyak sebuah *Predecessor* dan menuju sebuah *Successor*, berdasarkan penjelasan tentang *Triconnected Component* sebagaimana di jelaskan pada sub-bab 2.3.1.2 maka *Triconnected Abstraction* dapat di golongan menjadi 4 kelas yaitu *Trivial* (T), *Polygon* (P), *Bond* (B) dan *Rigid* (R), *Triconnected Abstraction* dapat digambarkan seperti di bawah ini:

a. *Trivial Abstraction*



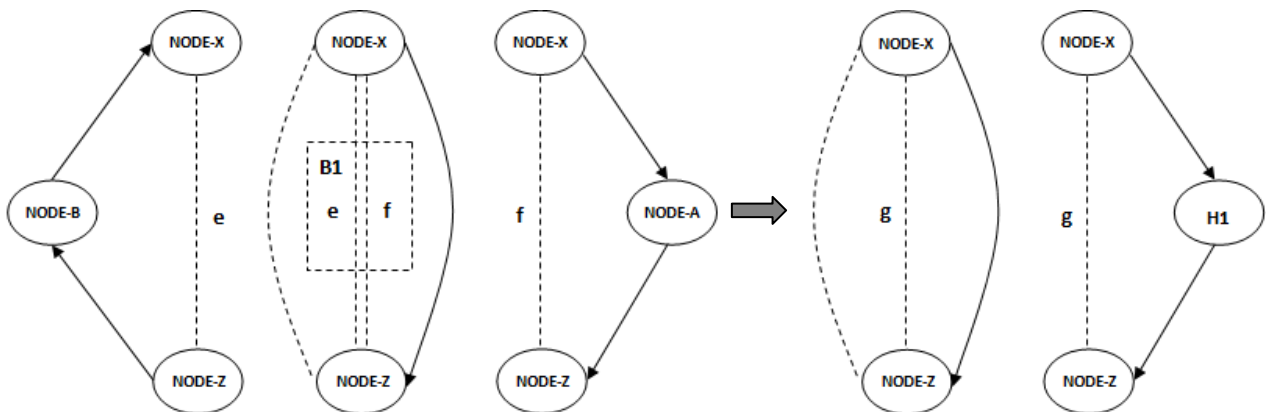
Gambar 2.5 Model Abstraksi dari Trivial

b. *Polygon Abstraction*



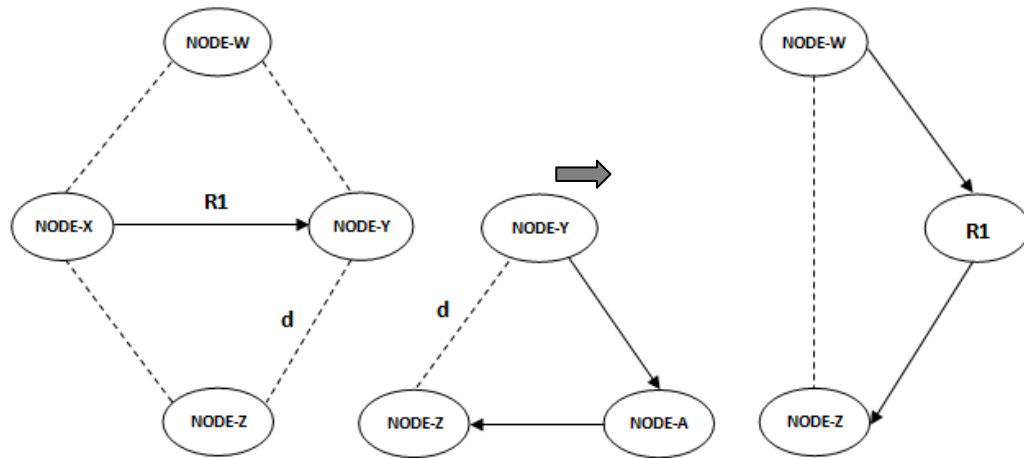
Gambar 2.6 Model Abstraksi dari Trivial

c. *Bond Abstraction*



Gambar 2.7 Model Abstraksi dari Bond

d. *Rigid Abstraction*



Gambar 2.8 Model Abstraksi dari Rigid

2.4 Analisis Kesamaan Proses Bisnis

Beberapa penelitian dalam melakukan analisis kesamaan dari proses bisnis salah satunya yang dilakukan oleh (Dijkman, Dumas, Dongen, Reina, & Mendling, 2011), mengemukakan bahwa untuk melakukan pengukuran tingkat kesamaan diantara proses model maka harus mengukur tingkat kesamaan di antara elemen, lima cara yang dapat dilakukan adalah dengan menganalisis tingkat kesamaan secara *sintaks*, semantik, atribut, tipe dan kontekstual dari proses model yang berbeda, pendekatan lain untuk melakukan analisis tingkat kesamaan menggunakan analisis secara struktural sebagaimana penelitian yang dilakukan oleh (Jung, Bae, & Liu, 2009) dengan menggunakan pendekatan *Process Vector*.

2.4.1 *Process Vector*

Berbagai studi dan pendekatan dalam melakukan analisis kesamaan dari proses bisnis salah satunya yang dilakukan berdasarkan definisi dari proses model, dimana proses model dibangun berdasar *Tuple* dimana $W = (A, T, Split \text{ dan } Join,)$ kemudian model proses bisnis diubah kedalam *Process Vector* di mana dalam *Process Vector* terdiri dari sekumpulan *activity* dan *transition*, dalam penelitian tersebut sekumpulan *activity* dan *transition* mencerminkan hubungan di antara kontrol aliran dari sebuah proses (*Control Flow*) di mana sebuah *transition* mempunyai ketergantungan terhadap *activity* sehingga dihitung *Execution Probability* berdasarkan pola kontrol aliran misalkan pola *Sequence* dan pola

percabangan (*OR*, *XOR* dan *AND*), sehingga bobot dari *Execution Probability* di antara sekumpulan *activity* sedangkan perhitungan *transition* dihitung melalui pendekatan pembobotan jarak berdasar *Weighted Completed Dependency Graph* (WCDG), sehingga perhitungan *Activity Vector* dihitung menggunakan Persamaan 2.1. dan *Transition Vector* dihitung menggunakan Persamaan 2.2.

$$\begin{aligned} a_x &= (a_{i,x}), a_{ix} = e_{i,x} \\ Pr(a) &= 1/s, \end{aligned} \quad (2.1)$$

di mana $i : 1, \dots, n$

| | | |
|-----------|---|---|
| a_x | : | <i>Execution Probability activity</i> , |
| $e_{i,x}$ | : | <i>Execution Probability</i> , |
| $Pr(a)$ | : | <i>Probability activity</i> , |
| s | : | Jumlah percabangan, |

$$t_x = (t_{ji,x}), t_{ix} = \frac{1}{d_{ijx}} e_{i,x} e_{j,x} \quad (2.2)$$

di mana $i : 1, \dots, n$

| | | |
|-----------|---|---|
| t_x | : | <i>Execution Probability transition</i> , |
| $t_{i,x}$ | : | <i>Distance Weight</i> , |

2.4.1.1 Matrik *Business Process Control Flow Complexity*

Sebuah representasi yang digunakan untuk menganalisis dan mengidentifikasi proses dan area yang kompleks dengan tujuan memberikan panduan agar proses dapat dibangun kembali (*reusable*).

2.4.1.2 Kompleksitas Sebuah Proses

Kompleksitas sebuah proses menurut (Cardoso, 2008) didefinisikan sebagai tingkat/kadar dari proses yang sulit untuk dianalisis, dimengerti atau dijelaskan yang ditandai oleh jumlah dan kerumitan dari *activity*, *transition*, percabangan, paralel, perulangan, kategori aktivitas, tipe dari struktur data dan karakteristik proses lainnya.

2.4.1.3 Pengukuran Kompleksitas Proses dalam Proses Bisnis

Perilaku *Control Flow Complexity* dibangun berdasar proses *Split* dan evaluasi kompleksitas dalam percabangan dibangun dari *OR-Split*, *XOR-Split* dan *And-Split* (Cardoso, 2008).

Tabel 2.1 Matrik *Control Flow Complexity* Proses Bisnis

| PERCABANGAN | RUMUS |
|-------------|---------------------------------------|
| <i>XOR</i> | $CFCXOR-split(a) = fan-out(a)$ |
| <i>OR</i> | $CFCOR-split(a) = 2^{fan-out(a)} - 1$ |
| <i>AND</i> | $CFCAND-split(a) = 1$ |

2.5 Model Ruang Vektor

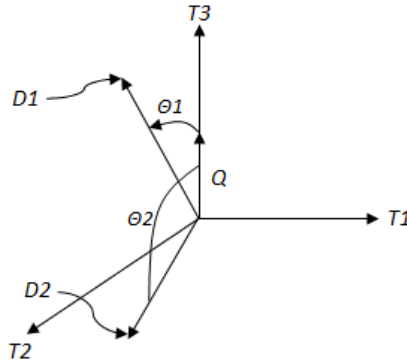
Model Ruang Vektor dapat digambarkan sebagai terdapat n kata yang berbeda sebagai kamus kata (*Vocabulary*) atau *Terms Index*. Ruang vektor yang dibentuk memiliki dimensi sebesar jumlah n kata, setiap i kata di dalam dokumen atau *Query* diberikan bobot sebesar w_i , baik dokumen maupun *Query* direpresentasikan sebagai vektor berdimensi n . Sebagai contoh terdapat 3 buah kata ($T1$, $T2$ dan $T3$), 2 buah dokumen ($D1$ dan $D2$) serta sebuah *Query* Q , masing-masing bernilai:

$$D1 = 2T1 + 3T2 + 5T3; D2 = 3T1 + 7T2 + 0T3; Q = 0T1 + 0T2 + 2T3$$

Representasi grafis dari ketiga vektor ini adalah seperti pada gambar koleksi dokumen direpresentasi pula dalam ruang vektor sebagai matrik kata dokumen (*Terms Documents Matrix*). Nilai dari elemen matrik w_{ij} adalah bobot kata i dalam dokumen j .

Penentuan relevansi dokumen dengan *Query* dipandang sebagai pengukuran kesamaan (*Similarity Measure*) antara vektor dokumen dengan vektor *Query*, semakin “sama” suatu vektor dokumen dengan vektor *Query* maka dokumen dapat dipandang semakin relevan dengan *Query*, salah satu pengukuran kesamaan yang baik adalah dengan memperhatikan perbedaan arah dari kedua vektor tersebut. Perbedaan arah kedua vektor dalam geometri dapat dianggap sebagai sudut yang terbentuk oleh kedua vektor. Gambar 3.4 mengilustrasikan kesamaan antara dokumen $D1$ dan $D2$ dengan *Query* Q . sudut $\theta1$ menggambarkan

kesamaan dokumen $D1$ dengan $Query$ sedangkan sudut θ_2 menggambarkan kesamaan dokumen $D2$ dengan $Query$.



Gambar 2.9 Representasi Vektor Dokumen dan $Query$

Q adalah vektor $Query$ dan D adalah vektor dokumen, yang merupakan dua buah vektor dalam ruang berdimensi- n , dan adalah sudut yang dibentuk oleh kedua vektor tersebut. Maka dengan $Q \cdot D$ adalah hasil perkalian titik (*Dot Product*) kedua vektor sebagaimana dijelaskan pada Persamaan 2.3.

$$Q \cdot D = |Q||D|\cos\theta \quad (2.3)$$

2.6 Pembobotan Kata (Term)

Bagian sebelumnya membahas mengenai metode untuk mengukur kesamaan di antara dokumen dan $Query$ dalam model ruang vektor. Dokumen maupun $Query$ direpresentasikan sebagai vektor berdimensi- n . Bagian ini akan membahas mengenai nilai dari vektor atau bobot kata dalam dokumen, salah satu cara untuk memberi bobot terhadap suatu kata adalah memberikan nilai jumlah kemunculan suatu kata (*Term Frequency*) sebagai bobot, semakin besar kemunculan suatu kata dalam dokumen akan memberikan nilai kesamaan yang semakin besar. Faktor lain yang diperhatikan dalam pemberian bobot adalah jarang munculnya kata (*Term Scarcity*) dalam koleksi. Kata yang muncul pada sedikit dokumen harus dipandang sebagai kata yang lebih penting (*Uncommon Terms*) dari pada kata yang muncul pada banyak dokumen. Pembobotan akan memperhitungkan faktor kebalikan frekuensi dokumen yang mengandung suatu kata (*Inverse Document Frequency*), faktor yang terakhir adalah faktor normalisasi terhadap panjang dokumen. Dokumen dalam koleksi dokumen memiliki karakteristik panjang yang beragam, ketimpangan terjadi karena

dokumen yang panjang akan cenderung mempunyai frekuensi kemunculan kata yang besar, sehingga untuk mengurangi ketimpangan tersebut diperlukan faktor normalisasi dalam pembobotan. Perbedaan antara normalisasi pada pembobotan dan perangkingan adalah normalisasi pada pembobotan dilakukan terhadap suatu kata dalam suatu dokumen sedangkan pada perangkingan dilakukan terhadap suatu dokumen dalam koleksi dokumen. Pembobotan suatu kata terhadap suatu dokumen dihitung menggunakan Persamaan 2.4.

$$w_i = \frac{\log(tf_i) + 1.0}{\sqrt{\sum_{j=1}^t [(\log(tf_i) + 1.0)]^2}} \quad (2.4)$$

sedangkan pembobotan suatu kata terhadap suatu *Query* dihitung menggunakan Persamaan 2.5.

$$w_i = \frac{\log(tf_i) + 1.0}{\sqrt{\sum_{j=1}^t [(\log(tf_i) + 1.0 \times (\log(N/n_i)))]^2}} \quad (2.5)$$

2.7 Latent Semantic Analysis (LSA)

Latent Semantic Analysis (LSA) adalah sebuah teori dan metode untuk mengekstraksi dan merepresentasikan arti sebuah kata-kata, arti dari sebuah kata diperkirakan dengan menggunakan metode perhitungan statistik yang diterapkan dalam kamus teks di mana sebuah kamus yang mengandung keterhubungan yang besar dalam menentukan kesamaan semantik dari sebuah kata terhadap kumpulan kata-kata, atau dapat diartikan sebagai metode yang digunakan untuk mencari dan menemukan informasi berdasarkan makna keseluruhan (*Conceptual Topic* atau *Meaning*) dari sebuah dokumen bukan hanya makna kata per kata, di mana *Query* diproses melalui operasi teks, kemudian vektor *Query* yang dibentuk dan dipetakan menjadi vektor *Query* terpetakan (*Mapped Query Vector*), dalam membentuk *Query* terpetakan, diperlukan hasil dekomposisi nilai *Singular* dari koleksi dokumen, pada koleksi dokumen, dilakukan operasi teks pada koleksi dokumen, kemudian matrik kata dokumen (*Terms Documents Matrix*) dibentuk, selanjutnya dilakukan dekomposisi nilai *Singular* (*Singular Value Decomposition*)

pada matrik kata-dokumen, hasil dekomposisi disimpan dalam *Collection Index*, proses *ranking* dilakukan dengan menghitung relevansi di antara vektor *Query* yang terpeta dengan *Collection Index*, proses pengindeksan dengan metode *Latent Semantic Analysis* menggunakan sebuah reduksi *Vector Spaces* untuk merepresentasikan kata/teks dan dokumen (Dumais, 1995), tahap reduksi ini dilakukan dengan mengkombinasikan informasi dari permukaan hingga lebih dalam dari sebuah dokumen untuk menentukan maksud yang sama di antara kata dalam dokumen (Magerman, Van Looy, Baesens, & Debackere, 2011).

2.7.1 *Singular Value Decomposition*

Singular Value Decomposition (SVD) merupakan sebuah teknik untuk mendekomposisi matriks berukuran apa saja yang diaplikasikan dalam matriks yang mempunyai ukuran yang sangat besar dengan tujuan untuk mempermudah pengolahan data, hasil dari SVD ini adalah *Singular Value* yang di simpan dalam sebuah matriks diagonal D berdasar urutan yang sesuai dengan keterhubungan vektor *Singular*, di mana nilai *Singular* menyimpan informasi yang sangat penting tentang data, yaitu data yang berkontribusi paling besar terhadap variasi data secara keseluruhan, yang di simpan pada singular value yang pertama, dalam penerapan SVD tidak terlepas dengan istilah tentang *Eigenvalue* dan *Eigenvector*.

2.7.2 *Eigenvalue dan Eigenvector*

Eigenvalue adalah sebuah bilangan skalar dan *Eigenvector* adalah sebuah matriks yang keduanya dapat mendefinisikan sebuah matriks, misalkan terdapat matrik A.

$$A = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 2 \end{bmatrix}$$

Gambar 2.10 Sebuah Matrik A

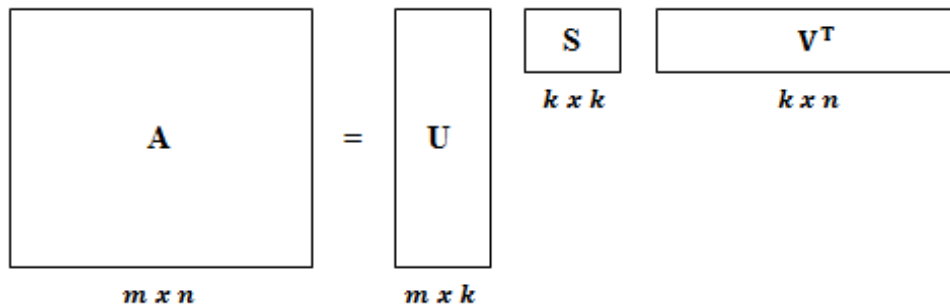
Representasi matrik pada Gambar 2.10 maka akan dapat diketahui nilai *Eigenvalue* dan nilai *Eigevector* adalah sebagai berikut:

$$\lambda_1 = 4 \quad X_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \lambda_2 = 3 \quad X_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad \lambda_3 = 2 \quad X_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Gambar 2.11 Nilai *Eigenvalue* dan *Eigenvector* dari Matrik A

2.7.3 Reduksi Dimensi

Pada penerapan SVD dalam pengindeksan dengan metode LSA menggunakan sebuah reduksi *Vector Spaces* untuk merepresentasikan kata/teks dan dokumen diperlukan sebuah reduksi dimensi untuk mengambil kumpulan objek yang terdapat pada ruang dimensi yang besar dan merepresentasikan kedalam ruang dimensi lebih yang kecil (Rosario, 2000). Reduksi dimensi ini di mungkinkan berdasarkan nilai singular k yang terbesar (Osmanli, 2010), pada Gambar 2.12. menggambarkan reduksi dimensi dari sebuah matriks.



Gambar 2.12 Reduksi Dimensi dari Sebuah Matrik

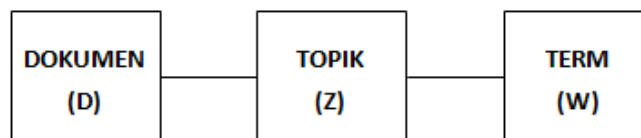
2.8 Probabilistic Latent Semantic Analysis (PLSA)

Tujuan dari pemodelan topik adalah menghasilkan representasi dokumen yang baik sehingga dapat di gunakan untuk berbagai macam task. (Hofmann, 1999) mengemukakan bahwa titik permulaan dari ide PLSA pada sebuah model statistik yang di sebut sebagai *Aspect Model* di mana sebuah dokumen teks terdiri dari kumpulan kata-kata, apabila ditarik satu hal di antara dokumen dan kata, maka akan dapat di peroleh kata kunci (*Keyword*) yang menjadi jembatan di antara dokumen dan kata, kata kunci ini yang di sebut sebagai *Aspect Model*. *Aspect Model* didefinisikan sebagai sebuah variabel yang tidak terlihat (*Latent Variable*) dari sebuah dokumen, semua variabel yang digunakan untuk

memodelkan *Aspect Model* tersebut melibatkan asumsi *Conditional Independence*. dan di representasikan menggunakan Persamaan 2.6.

$$P(d, w) = P(d)P(w|d), P(w|d) \sum_{e \in z} P(w|z)P(z|d) \quad (2.6)$$

Variabel di atas merupakan matriks probabilitas. Matrik yang digunakan pada PLSA tidak sama seperti *Term Document Matrix* yang ada pada LSA. Matriks yang digunakan pada PLSA sudah merupakan *Likelihood* dari faktor topik terkait dengan dokumen (*Document Given Topic*) dan juga terkait dengan kata (*Word Given Topic*), sedangkan kebalikan pada LSA, fungsi matriknya hanya berfungsi sebagai *Likelihood Term Frequency*.



Gambar 2.13 Hubungan Dokumen, Topik, dan Kata

Kehadiran Topik (Z) di antara Term (W) dan dokumen (D) sebenarnya menyatakan ketika berada pada dokumen (D), maka kata-kata yang muncul adalah Term (W), hal ini menunjukkan adanya sebuah ketergantungan (*Dependence*), namun karena ketergantungannya tidak secara langsung, sehingga hal ini dinamakan sebagai sebuah *Conditional Independence*.

Pembentukan model PLSA untuk menghasilkan kumpulan Topik (Z) beserta dengan nilai probabilitasnya diawali dengan pembuatan nilai probabilitas secara acak (*random*). Setelah diperoleh matriks awal probabilitas, matriks tersebut akan diproses ke dalam *training* dengan jumlah iterasi tertentu untuk memperoleh probabilitas yang terbaik dengan menggunakan algoritma *Expectation Maximization* (EM). Algoritma ini terdiri dari 2 tahap yaitu:

1) *Expectation* (E), di mana probabilitas *Posterior* dihitung untuk *Latent Variable*. Nilai probabilitas yang diperoleh dari langkah *Expectation* di hitung menggunakan Persamaan 2.7.

$$P(c_i|d_j) = \frac{P(c_i) \prod_{k_i}^{|d_j|} P(c_i)}{\sum_{r=1}^{|c|} P(c_i) \prod_{k_i}^{|d_j|} P(c_i)}, \quad (2.7)$$

di mana :

- $P(c_i)$: Probabilitas kemunculan kategori c_i ,
 $P(c_i|d_j)$: Probabilitas kemunculan kategori c_i dalam dokumen d_j ,
 $P(c_i) \prod_{k_i}^{|d_j|} P(c_i)$: Probabilitas kemunculan kategori c_i dalam kata k_i terhadap dokumen d_j .

2) *Maximization (M)*, di mana parameter yang ada akan diperbarui nilainya. Nilai yang di peroleh dari langkah Maximization di hitung menggunakan Persamaan 2.8.

$$P(w_{kj}|c_i) = \frac{\sum_{r=1}^{|D|} N(w_{kj}, d_j) p(c_i|d_j)}{\sum_{s=1}^{|w|} \sum_{j=1}^{|D|} N(w_s, d_j) p(c_i|d_j)}, \quad (2.8)$$

di mana :

- $N(w_{kj}, d_j)$: Jumlah kata w_k pada dokumen d_j .
 $|w|$: Jumlah keseluruhan kata/fitur yang di gunakan
 $|D|$: Jumlah seluruh *training* dokumen.

Algoritma EM bertujuan memperoleh nilai *error* yang semakin kecil, apabila nilai *error* masih tinggi, nilai bobot akan dioptimalkan supaya nilai *error* yang ada semakin kecil, semakin banyak *training*, maka topik serta probabilitas yang dibentuk akan semakin baik. Matriks yang dibentuk hasil dari PLSA terdiri dari $P(d|c)$, $P(w|z)$ dan $P(z)$. Matriks $P(d|z)$ adalah matriks probabilitas yang menggambarkan sebaran nilai topik pada sebuah dokumen, sedangkan matriks $P(w|z)$ menggambarkan nilai probabilitas topik pada setiap kumpulan kata-kata, lalu $P(z)$ merupakan nilai probabilitas dari topik itu sendiri.

$$P(d|z) = \begin{bmatrix} P d_1 | z_1 & & & & \\ & & & & \\ & & \dots & & \\ & & & & P d_n | z_k \\ & & & & \end{bmatrix} \begin{matrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ \dots \\ d_n \end{matrix}$$

$z_1 \quad z_2 \quad z_3 \quad z_4 \quad \dots \quad z_k$

Gambar 2.14 Matrik $P(d|z)$

Skema matriks di atas, memperlihatkan hubungan antara baris dan kolom yaitu bahwa pada setiap dokumen akan memiliki probabilitas topik masing-masing, sebagai contoh d_1 (dokumen pertama) memiliki probabilitas dari z_1 (topik pertama), z_2 , hingga z_n , begitu juga dengan d_2 , d_3 , d_4 , dan seterusnya. Data yang digunakan untuk diinputkan ke PLSA adalah dokumen yang menggunakan representasi berupa *Term Document Matrix*, misalnya dimensi awal dari *Term Document Matrix* adalah n buah dokumen, dan m buah kata, apabila kita memiliki 1.000 dokumen dan 10.000 kata, maka komponen dari *Term Document Matrix* akan berjumlah 10.000.000, ukuran 1.000×10.000 , dapat dibuat distribusi probabilitas $P(d|w)$, contoh, dari distribusi data tersebut, apabila hendak diketahui berapa jumlah kata “check” pada dokumen pertama, maka dapat diperoleh dari sel perpotongan antara dokumen pertama (pada kolom pertama matrik) dengan kata “check” (pada baris tertentu pada matrik), hal ini membuat pengolahan data terlihat praktis dan hasilnya pun pasti baik dan relevan, namun permasalahan yang terjadi pada dimensi 1.000×10.000 tersebut adalah dimensi matrik yang terlalu besar, dimensi yang terlalu besar akan memperlambat proses komputasi dan banyak proses yang tidak perlu dilakukan akan tetap dilakukan. Matrik yang terlalu besar akan mengakibatkan pemborosan memori di dalam prosesnya. PLSA menjadi salah satu solusi reduksi dimensi. Hal ini terjadi karena PLSA memiliki jembatan berupa topik Z yang membagi probabilitas menjadi 3 matrik tetapi dengan dimensi yang lebih kecil, pada proses reduksi dimensi ini, *Singular Value Decomposition* (SVD) akan sangat berperan.

2.9 Metode Klasifikasi Naïve Bayes

Algoritma *Naïve Bayes* merupakan algoritma klasifikasi menggunakan model probabilistik dan statistik yang di kemukakan oleh *Thomas Bayes*. *Teorema Naïve Bayes* ini adalah sebuah kombinasi antara *Teorema Naïve* dan *Bayes* Pada algoritma ini akan melakukan prediksi peluang di masa depan berdasarkan data sebelumnya yang dikenal dengan *Teorema Bayes*, sedangkan *Naïve* mengasumsikan bahwa kondisi antar atribut saling bebas atau tidak terkait. Sebagaimana menurut klasifikasi *Naïve Bayes* diasumsikan bahwa ada atau tidak ciri tertentu dari sebuah kelas tidak ada hubungannya dengan ciri dari kelas lainnya, sebagaimana pada Persamaan 2.9.

$$P(H|X) = \frac{P(H|X) \times P(H)}{P(X)}, \quad (2.9)$$

di mana:

| | | |
|----------|---|---|
| X | : | Data dengan <i>class</i> yang belum diketahui |
| H | : | Hipotesis data X merupakan suatu kelas spesifik |
| $P(H X)$ | : | <i>Probabilitas</i> hipotesis H berdasar kondisi X (<i>Posteriori Probability</i>) |
| $P(H)$ | : | <i>Probabilitas</i> hipotesis H (<i>Prior Probability</i>) |
| $P(X H)$ | : | <i>Probabilitas</i> X berdasarkan kondisi pada hipotesis H |
| $P(X)$ | : | <i>Probabilitas</i> X |

Klasifikasi *Naïve Bayes* memerlukan sejumlah data sebelunya untuk menentukan prediksi kelas yang sesuai dari sampel yang sedang dianalisis sehingga teorema *Naïve Bayes* sesuai dengan Persamaan 2.10.

$$P(C|F_1 \dots F_n) = \frac{P(C)P(F_1 \dots F_n|C)}{P(F_1 \dots F_n)}, \quad (2.10)$$

di mana :

| | | |
|-------|---|---|
| C | : | Merupakan representasi dari kelas |
| F_n | : | Merupakan karakteristik untuk melakukan klasifikasi |

Dari persamaan di atas dapat dijelaskan bahwa peluang dari karakteristik tertentu dalam kelas C (*Posterior*) adalah peluang munculnya kelas C (sebelum masuknya sampel tersebut, yang disebut *Prior*), dikali dengan peluang kemunculan karakteristik–karakteristik sampel pada kelas C (di sebut juga *Likelihood*) (Bustami, 2014).

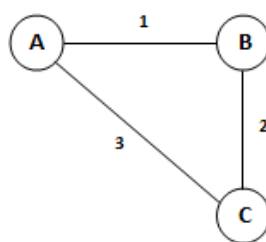
2.10 *Weighted Directed Acyclic Graph* (WDAG)

Dalam pembahasan ini bertujuan untuk menjelaskan tentang pengertian WDAG yang nantinya di gunakan sebagai representasi yang di gunakan untuk menentukan perhitungan kesamaan gabungan.

2.10.1 Teori *Graph*

Sebuah *graph* merupakan sebuah pemodelan yang banyak digunakan dalam berbagai hal.

- ***Graph*** terdiri dari $G = (V, E)$ di mana:
 - **V** : kumpulan simpul (*Vertices/Nodes*)
 - **E** : kumpulan sisi/busur (*Edge*) yang menghubungkan simpul-simpul
 - Sebuah sisi/busur (*Edge*) = (A, B) memiliki informasi dua simpul yang di hubungkannya



Gambar 2.15 Contoh dari *Graph*

- Istilah-istilah dalam *Graph*:
 - *Undirected graph* adalah *graph* yang tidak mempunyai arah.
 - *Directed graph* adalah *graph* yang mempunyai arah.
 - *Degree* (dari sebuah *vertex*) adalah jumlah simpul lain yang terhubung langsung melalui sebuah sisi.

- *Weighted Graph* adalah setiap sisi memiliki bobot/nilai.
- Jalur/*Path* adalah urutan simpul (*Vertices*) v_1, v_2, \dots, v_k sedemikian sehingga simpul yang berurutan v_k dan v_{k+1} adalah simpul yang terhubung.
- *Simple Path* adalah tidak ada simpul yang di ulang.
- *Cycle* adalah *Simple Path*, dengan catatan simpul awal sama dengan simpul akhir.
- DAG (*Directed Acyclic Graph*) adalah *graph* dengan busur/sisi yang memiliki arah dan tidak memiliki *Cycle*.
- *Connected Graph* adalah tiap simpul terhubung dengan simpul lain.
- Sub-*graph* adalah bagian simpul dan sisi yang dapat membentuk *graph*.

2.10.2 Perhitungan Kesamaan *Weighted Directed Acyclic Graph* (WDAG)

Teori dan istilah-istilah tentang *graph* di atas, maka dapat diketahui bahwa sekumpulan *graph* yang memiliki arah dalam setiap busur/sisi nya serta setiap busur/sisinya mempunyai label dan berarah merupakan pengertian dari WDAG, pada sub-bab ini menjelaskan algoritma perhitungan kesamaan dari WDAG, berdasarkan penelitian yang di lakukan oleh (Jin, 2006) algoritma kesamaan dari WDAG bentuk dasar dipengaruhi oleh:

1. *WDAG Similarity*
2. *WDAG Simplicity*

Persamaan 2.11 yang menjadi dasar untuk perhitungan kesamaan WDAG:

$$WDAGsim(g, g'): \quad (2.11)$$

$$\left\{ \begin{array}{ll} 0.0 & \text{Jika } root \text{ node label dari } g \text{ dan } g' \text{ tidak identik} \\ 1.0 & \text{Jika } g \text{ dan } g' \text{ adalah } leaf \text{ node} \\ \sum \left\{ \begin{array}{l} WDAGsim(g_i, g'_i) \cdot \frac{(w_i + w'_i)}{2} \\ WDAGsim(g_i, \varepsilon) \cdot \frac{(w_i + 0)}{2} \\ WDAGsim(\varepsilon, g'_j) \cdot \frac{(0 + w'_j)}{2} \\ \sum_{j=1}^{breadth \text{ of } g'} WDAGsim(\varepsilon, g'_j) \cdot \frac{(0 + w'_j)}{2} \\ \sum_{i=1}^{breadth \text{ of } g} WDAGsim(g_i, \varepsilon) \cdot \frac{(w_i + 0)}{2}, \end{array} \right. \end{array} \right.$$

di mana:

- $WDAGsim(g, g')$ adalah kemiripan antara dua buah WDAG g dan g'
- $WDAGsim(g_i, g'_i)$ adalah nilai kemiripan antara sub WDAG ke- i dan ke- j
- w_i dan w_j adalah bobot/weight dari turunan node dua buah WDAG g dan g' , sedangkan ε adalah WDAG kosong.
- i adalah nilai *index* yang meningkat dari 1 hingga nilai *breadth* dari g .
- j adalah nilai *index* yang meningkat dari 1 hingga nilai *breadth* dari g' .

Nilai *Simplicity* dari sebuah WDAG adalah [0,1] yang akan berkurang dengan meningkatnya *depth* (kedalaman) dan *breadth*. Semakin sederhana sebuah WDAG, maka semakin tinggi nilai *Simplicity* yang dimilikinya. Persamaan yang di gunakan oleh (Jin, 2006) dalam menentukan nilai *Simplicity* akan hilangnya *subWDAG* dari $WDAG(g)$ atau dalam rumus dijelaskan menggunakan istilah $WDAGplicity(g)$ Persamaan 2.12 adalah sebagai berikut:

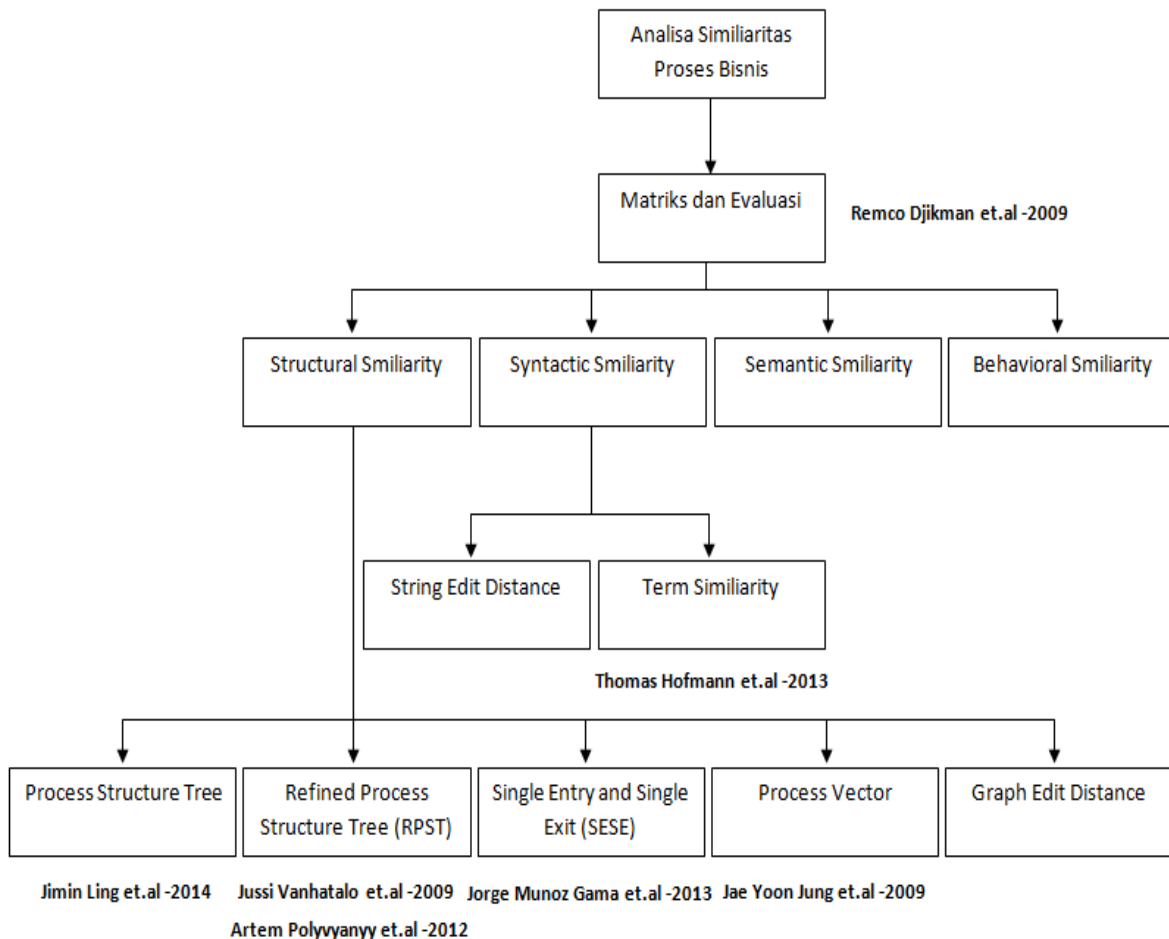
$$WDAGplicity(g) = \begin{cases} D_i & \text{Jika root node label dari } g \text{ dan } g' \text{ tidak identik} \\ \frac{D_f}{m} \sum_{j=1}^{|D|} WDAGsim(g_j) & \text{untuk } g \text{ selain leaf node} \end{cases} \quad (2.12)$$

di mana:

- D_i adalah *Depth Degradation Index*.
- D_f adalah *Depth Degradation Factor*
- m adalah *breadth* dari $WDAG(g)$ untuk $WDAG(g)$ bukan *leaf*.
- Jika g dan g' bukan merupakan *node* pada *leaf* dan memiliki *arc label* yang berbeda, maka nilai kemiripan (*similarity*) adalah 0, sebaliknya jika *arc label* yang dimiliki adalah sama maka akan dilakukan *Recursive Traversal Top-Down*, melalui *subWDAG* yang di miliki, selama proses *Travesal*, apabila g dan g' memiliki *arc label* yang sama akan dilakukan perhitungan nilai kesamaan $WDAG(g_i, g_j')$ seperti pada Persamaan 2.11, akan tetapi jika terdapat g_i atau g_j' yang hilang dalam *subWDAG* yang di lalui, maka nilai kemiripan dari *subWDAG* yang hilang adalah berupa nilai *simplicity* dikalikan 0.5, atau $WDAGsim(null, g_j') = WDAGplicity(null, g_j') \times 0.5$, oleh karena nilai kemiripan dari g dan g' , adalah jumlah dari kemiripan seluruh *subWDAG* g_i dan g_j' yang memiliki *arc label* identik serta nilai kemiripan dari seluruh *subWDAG* yang hilang/*null*.

2.11. Penelitian Terkait

Beberapa penelitian yang terkait dalam penelitian ini akan dijelaskan pada Gambar 2.16.



Gambar 2.16 Penelitian Terkait

Penelitian terkait yang telah dilakukan dan menjadi latar belakang dari penelitian yang kami lakukan. Penelitian yang dilakukan oleh (Dijkman et al., 2011) tentang matrik dan evaluasi kesamaan proses bisnis, beberapa pendekatan dapat dilakukan dalam melakukan analisis kesamaan, di antaranya menggunakan pendekatan struktural, *sintaks*, semantik dan behavioral, dalam penelitian yang kami lakukan, kami menggabungkan pendekatan struktural dan secara sintaksis semantik, di mana acuan yang kami gunakan dalam pendekatan struktural adalah penelitian dari (Vanhatalo et al., 2009) dan kemudian dikembangkan oleh

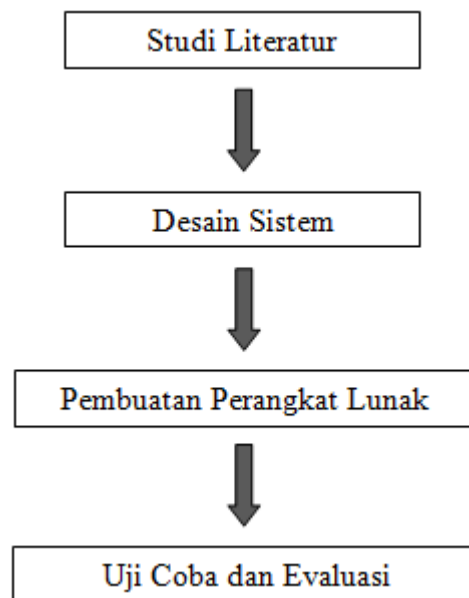
(Polyvyanyy, 2012), tentang dekomposisi *Refined Process Structured Tree* (RPST), dan *Abstraction Model* menggunakan *Triconnected Abstraction* sebagai representasi *Weighted Directed Acyclic Graph* (WDAG), dan untuk pendekatan analisis secara sintaksis semantik kami menggunakan penelitian dari (Hofmann, 1999) tentang *Probabilistic Latent Semantic Analysis* (PLSA).

(halaman ini sengaja dikosongkan)

BAB III

METODOLOGI PENELITIAN

Pada bab ini akan dijelaskan Metodologi Penelitian pada Penelitian ini. Langkah-langkah yang akan dilakukan pada penelitian ini meliputi (1) Studi Literatur, (2) Desain Sistem, (3) Pembuatan Perangkat Lunak, (4) Uji Coba dan Evaluasi. Tahapan metodologi Penelitian dapat dilihat pada Gambar 3.1.



Gambar 3.1 Tahapan Metodologi Penelitian

3.1 Studi Literatur

Tahap studi literatur, dikaji berbagai referensi yang berkaitan dengan pemodelan proses bisnis, memecah model proses, analisis kesamaan di antara model proses secara struktural dan semantik, representasi model proses kedalam bentuk *graph* dan algoritma perhitungan kesamaan dan perluasan dari model proses. Studi literatur yang dilakukan diharapkan dapat memberikan gambaran secara lengkap dan dapat memberikan dasar kontribusi tentang analisis tingkat kesamaan suatu proses model yang akan dilakukan pada penelitian ini. Studi *literatur* yang dilakukan adalah mempelajari beberapa referensi sebagai berikut:

1. Penelitian terdahulu tentang analisis kesamaan model proses menggunakan analisis secara struktural dan tekstual.

2. Dasar teori tentang kesamaan elemen model proses
3. Metode pemetaan *Process Vector*
4. Metode *Probabilistic Latent Semantic Analysis* (PLSA)
5. Metode *Weighted Directed Acyclic Graph* (WDAG)

3.2 Desain Sistem

Pada sub-bab ini akan dibahas perancangan solusi yang diusulkan. Fitur utama yang disediakan dari solusi ini adalah sebagai berikut:

1. Metode analisis gabungan menggunakan analisis secara struktural dan analisis *tekstual* (semantik sintaks)
2. Representasi analisis gabungan kedalam *Weighted Directed Acyclic Graph* (WDAG).
3. Perhitungan kesamaan di antara model proses yang dibandingkan.

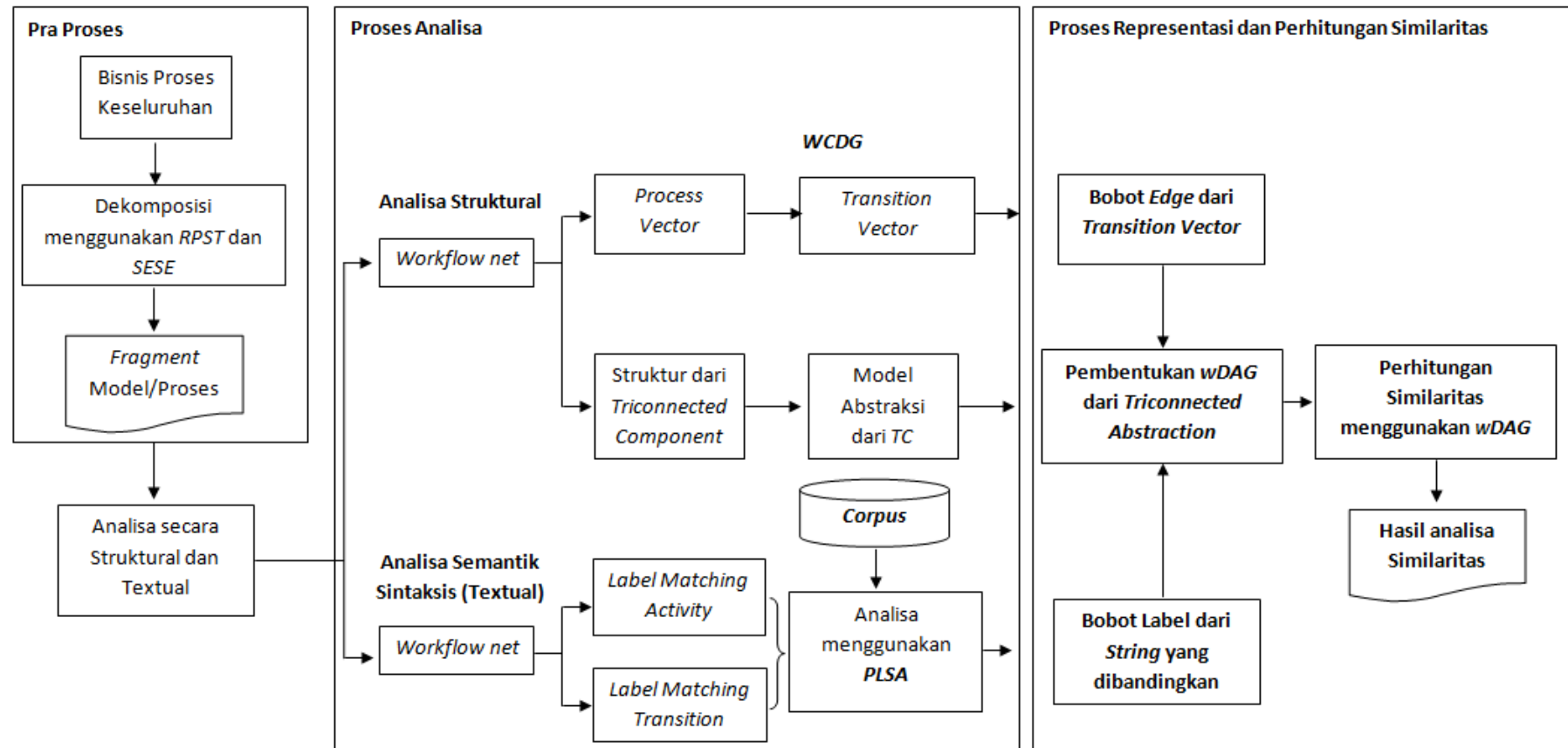
Secara umum desain sistem yang diajukan dapat dilihat pada Gambar 3.2.

3.2.1 Pengumpulan Dataset

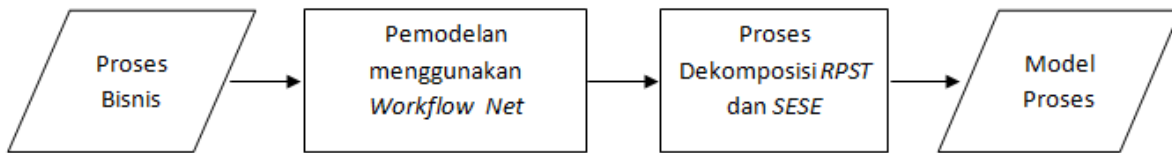
Pada tahap ini dilakukan untuk mengambil data proses bisnis PPDB yang kemudian dimodelkan dan didekomposisi menggunakan metode *Refined Process Structured Tree* (RPST) dan *Single Entry and Single Exit* (SESE) menjadi sub proses atau model *fragment*.

3.2.2 Pra-Proses

Pada tahap pra proses proses bisnis Penerimaan Peserta Didik Baru secara online (PPDB) dimodelkan menggunakan *Workflow Net* kemudian dilakukan proses dekomposisi menggunakan metode *Refined Process Structured Tree* (RPST) dan *Single Entry and Single Exit* (SESE) menjadi sub-proses atau model proses, dimana model proses ini yang akan dilakukan analisis untuk mendapatkan nilai kesamaan diantara model proses yang dibandingkan. Gambar 3.3 merupakan alur pra-proses.



Gambar 3.2 Desain Sistem yang Diajukan



Gambar 3.3 Alur Pra-Proses

3.2.2.1 Proses Bisnis Menggunakan Pemodelan *Workflow Net*

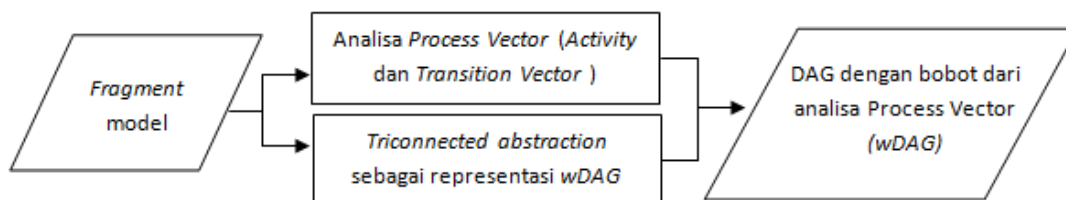
Gambar 3.5 adalah model *fragment Pre Registration* yang dimodelkan menggunakan *Workflow Net* sebagaimana dijelaskan pada sub-bab 2.1 dimana *Workflow Net* direpresentasikan berupa *Node* dan *Edge*.

3.2.2.2 Dekomposisi RPST dan SESE

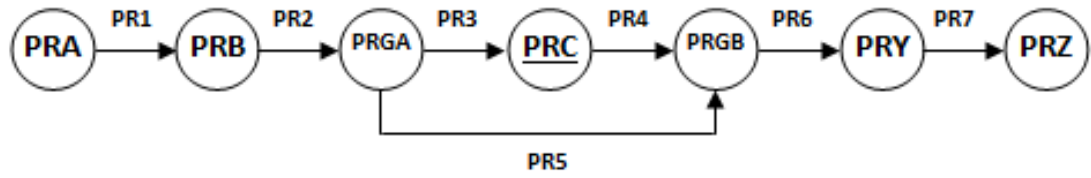
Tahap ini dilakukan penguraian atau memecah proses bisnis pada Gambar 3.5 menjadi beberapa sub-proses untuk disimpan kedalam repositori. Penguraian ini dilakukan menggunakan metode RPST dan SESE sebagaimana dijelaskan pada sub-bab 2.2.1 hasil dari penguraian dapat dilihat pada Gambar 3.6 merupakan hasil dekomposisi dari *fragment* model *Pre Registration* 1 dan 3, sedangkan Gambar 3.7 merupakan hasil RPST dari *fragment* model *Pre Registration*.

3.2.3 Analisis Struktural

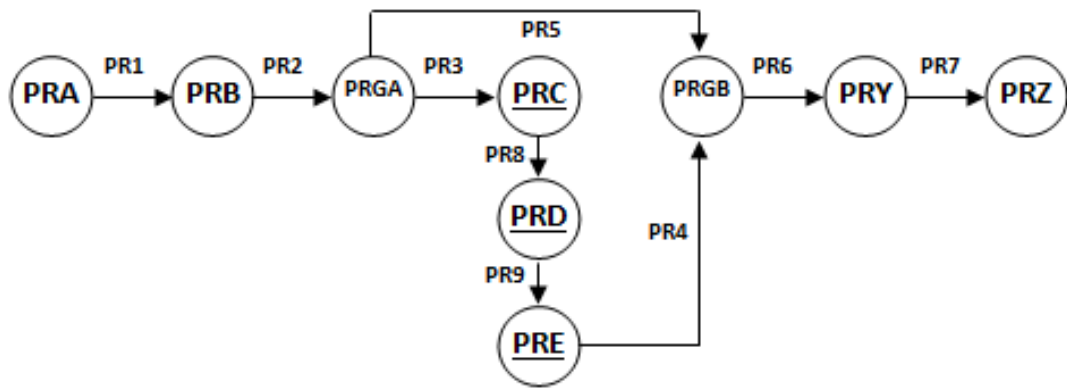
Analisis Struktural dilakukan setelah proses bisnis keseluruhan didekomposisi menjadi sub-proses, proses berikutnya adalah melakukan analisis secara struktural untuk mendapatkan nilai kesamaan struktural diantara proses model yang dibandingkan, tahap untuk melakukan analisis kesamaan secara struktur dimulai dengan menganalisis *Process Vector*, alur analisis secara struktural pada Gambar 3.4.



Gambar 3.4 Alur Analisis Struktural

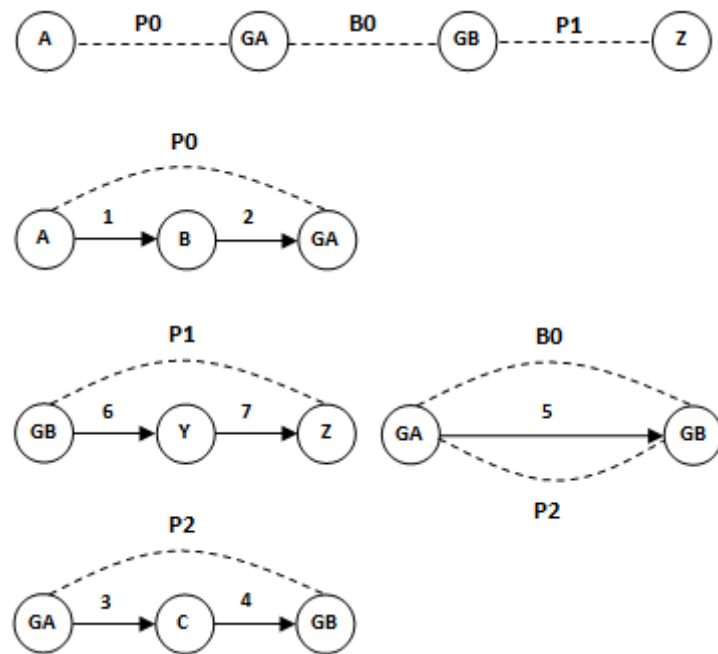


(a)

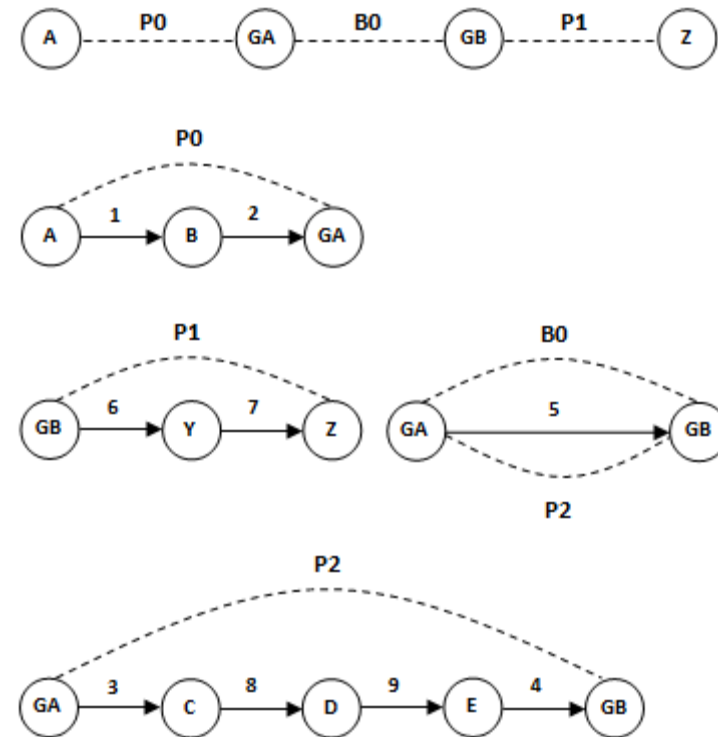


(b)

Gambar 3.5 (a) Model *Pre Registration 1*, (b) Model *Pre Registration 3*

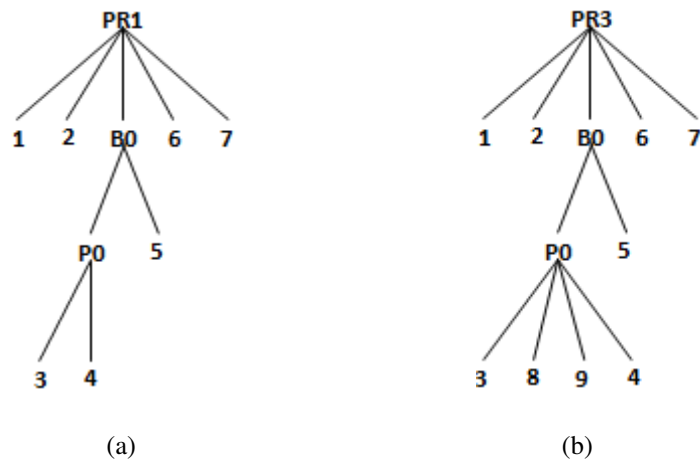


(a)



(b)

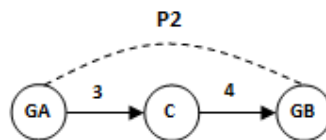
Gambar 3.6 Contoh Hasil Dekomposisi dari (a) Model *Fragment* dari *Pre Registration 1*, (b) Model *Fragment* dari *Pre Registration 3*



Gambar 3.7 (a) RPST *Fragment Pre Registration* 1, (b) RPST *Fragment Pre Registration* 3

3.2.3.1 Process Vector

Tahap ini digunakan untuk menentukan nilai/bobot dari label dalam WDAG, nilai didapatkan dari perhitungan *Transition Vector* dari sebuah proses model, pada penelitian sebelumnya (Jung, Bae, & Liu, 2009), *Process Vector* merupakan kumpulan dari *Activity Vector* dan *Transition Vector* yang mencerminkan interaksi dalam sebuah model proses. Perhitungan *Transition Vector* dilakukan dengan pendekatan *Weighted Completed Dependency Graph* (WCDG). Perhitungan berdasarkan rumus pada Persamaan 2.1 dan Persamaan 2.2 sebagai contoh pada Tabel 3.1 dari model *fragment* pada Gambar 3.8.



Gambar 3.8 Model *Fragment* P2 Model *Pre Registration* 1

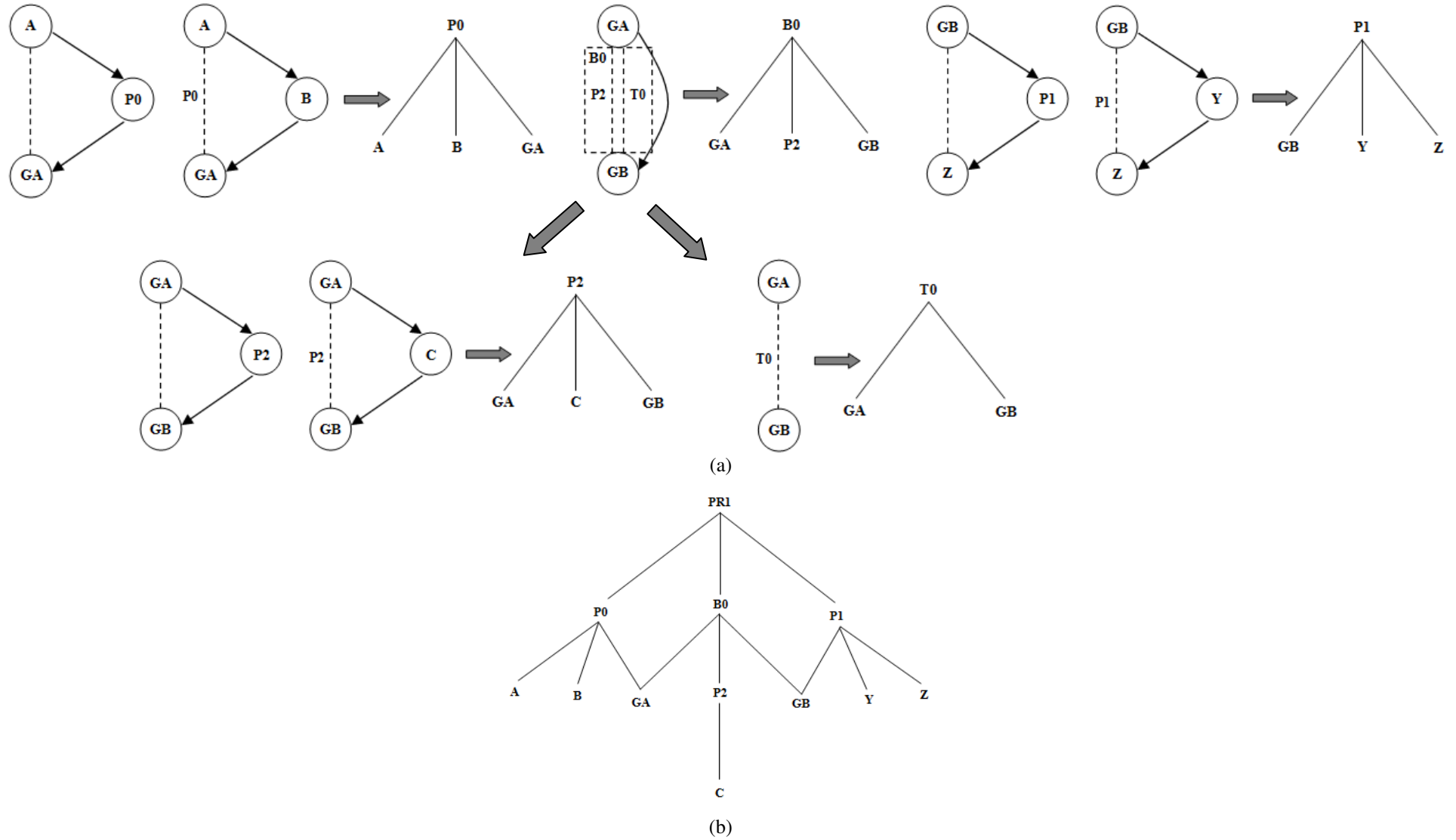
Tabel 3.1 *Process Vector* Model *Fragment* P2

| PROSES2_PR | | PEMBOBOTAN | | | |
|------------|---------------------------|------------|----------|-------|------------|
| NODE | LABEL STRING | VALUE NODE | DISTANCE | BOBOT | BOBOT WDAG |
| GA | ACQUIRE REWARD | 1 | | 1 | 0,4 |
| C | CALL ACHIEVEMENT STUDENT | 0,5 | 1 | 0,5 | 0,2 |
| GB | JOIN STUDENT VERIFICATION | 1 | | 1 | 0,4 |
| | | | | 2,5 | 1 |

Tabel 3.1. menunjukan sub-proses yang membangun model *fragment* P2, diantaranya adalah “*acquire reward*”, “*call achievement student*”, “*join student verification*”, dalam menentukan bobot WDAG setiap sub-proses dihitung *Execution Probability activity* terlebih dahulu berdasarkan Persamaan 2.1, kemudian dihitung nilai *Execution Probability transition*, misalnya, pada sub-proses “*call achievement student*” pada kolom *value node* dihitung berdasarkan *Execution Probability activity* maka hasilnya adalah 0,5, hasil ini dikarenakan node “*call achievement student*” berada dalam percabangan dimana *Node Entry* adalah “*acquire reward*” yang memiliki 2 *Successor* yaitu Node “*call achievement student*” dan Node “*join student verification*”, sedangkan untuk *Execution Probability transition* dihitung menggunakan Persamaan 2.2, dimana *Distance Weight node “call achievement student”* bernilai maka nilai *Execution Probability transition* adalah $0,5 \times 1 = 0,5$, agar hasil dari bobot WDAG ≤ 1 maka dilakukan normalisasi terhadap bobot *Proces Vector*.

3.2.3.2 *Triconnected Abstraction*

Perhitungan kesamaan gabungan antara analisis secara struktural dan secara *tekstual* diperlukan representasi kedalam *graph* yang mempunyai label dan berbobot. Penelitian ini untuk merepresentasikan sebuah proses model kedalam bentuk WDAG, kami menggunakan *Abstraction Model* yang dikemukakan oleh (Polyvyanyy, 2012) model *Triconnected Component*, sebagaimana dijelaskan pada sub bab 2.3.1, dimana hasil dari dekomposisi *Refined Process Structured Tree* (RPST) dan SESE merupakan *fragment* model *Triconnected Component* sehingga hasil dekomposisi tersebut dapat direpresentasikan kedalam WDAG dengan menggunakan model abstraksi, sebagaimana contoh pada Gambar 3.9 merupakan representasi WDAG dari *fragment Pre Registration* 1 dan 3.



Gambar 3.9 Triconnected Abstraction (a) Fragment Pre Registration 1, (b) WDAG Pre Registration 1

3.2.4 Analisis Tekstual (Semantik *Sintaks*)

Setelah dilakukan analisis secara struktural pada tahap ini akan dilakukan analisis secara tekstual, analisis tekstual dilakukan dengan memeriksa kesamaan arti kata dalam label dari setiap *activity* dan *transition* yang ada dalam model proses. Pada Tabel 3.2 dan 3.3 diketahui bahwa setiap *activity* mempunyai *label string* yang merupakan nama atau keterangan dari *activity* sebuah proses bisnis nilai dari analisis struktural belum tentu menunjukkan kesesuaian dari setiap model proses, salah satu permasalahan saat melakukan pencocokan label *string* adalah permasalahan pemeriksaan sinonim.

Tabel 3.2 Label String Node Model Pre Registration 1

| <i>Node</i> | <i>Label String</i> |
|-------------|---------------------------|
| B | Check Student Achievement |

Tabel 3.3 Label String Node Model Pre Registration 3

| <i>Node</i> | <i>Label String</i> |
|-------------|--------------------------|
| B | Verify Student Accession |

Tabel 3.2 dan 3.3 di atas menunjukkan label *string* dari *activity* yang ditunjukkan dengan *Node B* dari *fragment Pre Registration 1* dan *node B fragment Pre Registration 3*, untuk mengukur nilai kesamaan dari 2 buah label *string* dibandingkan dapat menggunakan metode *Cosine Measure*, dimana nilai kesamaan ditunjukkan menggunakan interval nilai 0-1, nilai mendekati 1 menunjukkan semakin identik, sebagai contoh untuk membandingkan nilai kesamaan label *string* pada *Node B* terdapat pada Tabel 3.4.

Tabel 3.4 Perhitungan Cosine Measure Label String Node B

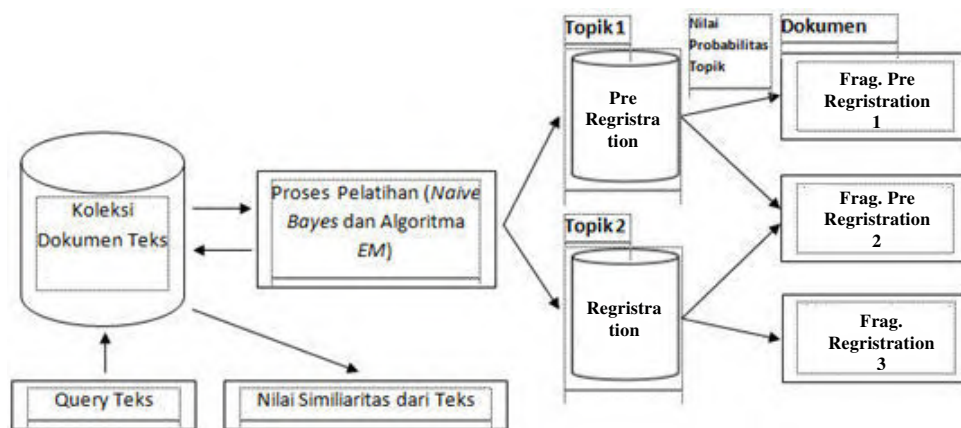
| <i>Activity</i> | <i>Label String</i> | <i>Check</i> | <i>Verify</i> | <i>Student</i> | <i>Achievement</i> | <i>Accession</i> |
|-----------------|----------------------------------|--------------|---------------|----------------|--------------------|------------------|
| <i>Node B</i> | <i>Check Student Achievement</i> | 1 | 0 | 1 | 1 | 0 |
| <i>Node B</i> | <i>Verify Student Accession</i> | 0 | 1 | 1 | 0 | 1 |

$$\text{Cosine Measure} = \frac{(1 \times 0) + (0 \times 1) + (1 \times 1) + (1 \times 0) + (0 \times 1)}{\sqrt{((1^2) + (0^2) + (1^2) + (1^2) + (0^2))} \times \sqrt{((0^2) + (1^2) + (1^2) + (0^2) + (1^2))}} \\ = 0.33333$$

Perhitungan menggunakan *Cosine Measure* didapatkan nilai kesamaan dari label *string Node B* adalah **0.33333**, menandakan bahwa nilai tersebut mempunyai keterhubungan yang rendah, namun apabila kita melihat arti/makna dari label *Node B* mempunyai makna yang sama, seharusnya nilai dari kesamaan label *string* adalah mendekati 1.

3.2.4.1 Analisis Tekstual PLSA

Latar belakang diatas diperlukan sebuah pendekatan untuk menghitung nilai kesamaan dari label *string* yang lebih baik, dan pada penelitian ini kami menggunakan *Probabilistic Latent Semantic Analysis* (PLSA). Gambar 3.10 menunjukkan alur analisis *tekstual* menggunakan PLSA. Tahap pertama dalam melakukan analisis *tekstual* adalah dengan membangun koleksi dokumen di mana sebelumnya dilakukan tahap pra-proses untuk mendapatkan bentuk yang mewakili makna dari dokumen secara utuh, tahap ini dilakukan proses *tokenisasi*, pembuangan *stop word* agar ketika kita melakukan *training* untuk mendapatkan nilai probabilitas yang baik, langkah berikutnya adalah melakukan training untuk mendapatkan nilai probabilitas yang baik menggunakan algoritma EM sebagaimana dijelaskan pada sub-bab 2.8, namun sebelum menggunakan algoritma EM terlebih dahulu kita lakukan proses klasifikasi dari dokumen untuk menentukan topik dan nilai probabilitas dari topik tersebut.



Gambar 3.10 Alur Analisis Tekstual PLSA

3.2.4.1.1 Metode Klasifikasi *Naïve Bayes*

Sebagaimana dijelaskan tentang teori klasifikasi *Naïve Bayes* pada sub-bab 2.9. *Naïve Bayes* membangun model probabilitas dari matrik *term* dokumen dari data yang berlabel.

Tabel 3.5 Tabel Contoh Prediksi Kategori

| Dokumen | Kategori | Fitur (Kemunculan) |
|-----------|-------------------------|---|
| Dokumen 1 | <i>Pre Registration</i> | <i>Achieve (4), Call(4), Check(2), Exam(5), Health(1), Print(4), Registration (4), Student (2), Verify (2)</i> |
| Dokumen 2 | <i>Registration</i> | <i>District (1), Enter(3), ID(3), Limited (1), Print (3), Registration (14), School (3), Select (3), Student (3), Unlimited (1), Verify (3)</i> |
| Dokumen 3 | ? | <i>Enter (1), ID (1), Print (1), Registration (5), School (1), Select (1), Student (2), Verify (1)</i> |

Dari kumpulan dokumen pada Tabel 3.5, kemudian akan dibentuk matrik *term* dokumen sebagaimana ditunjukkan pada Tabel 3.6, langkah selanjutnya adalah pembuatan model probabilitas dengan melakukan perhitungan menggunakan Persamaan 3.1, kemudian langkah terakhir menentukan kategori untuk dokumen 3.

$$p(w_{kj}|c_i) = \frac{f(w_{kj}|c_i) + 1}{f(c_i) + |w|}, \quad (3.1)$$

di mana :

- $f(w_{kj}|c_i)$: Nilai kemunculan kata w_{kj} pada kategori c_i .
- $f(c_i)$: Jumlah keseluruhan kata pada kategori c_i .
- $|w|$: Jumlah keseluruhan kata/fitur yang digunakan

dan
$$p(c_i) = \frac{fd(c_i)}{|D|}, \quad (3.2)$$

di mana :

- $fd(c_i)$: Jumlah dokumen yang memiliki kategori c_i .
- $|D|$: Jumlah seluruh *training* dokumen.

didapat dari Persamaan 3.2, sehingga model probabilitas yang terbentuk pada Tabel 3.7, berikut contoh perhitungan menggunakan Persamaan 3.2.

- $$\begin{aligned}
 p("pre"|"d3") &= p("pre") \times p("verify"|"pre") \times \\
 &\quad p("registration"|"pre") \times p("student"|"pre") \\
 &= 1/2 \times 3/25 \times 5/25 \times 5/25 \\
 &= \mathbf{0,0024}
 \end{aligned}$$

- $$\begin{aligned}
 p("reg"|"d3") &= p("reg") \times p("verify"|"reg") \times \\
 &\quad p("registration"|"reg") \times p("student"|"reg") \\
 &= 1/2 \times 4/27 \times 15/27 \times 4/27 \\
 &= \mathbf{0,006096632}
 \end{aligned}$$

- Karena $p("reg"|"d3") > p("pre"|"d3")$, maka kategori dari dokumen3 adalah **Registration**.

3.2.4.1.2 Metode *Expectation Maximization*

Tahap berikutnya dalam fase *training* dari metode PLSA, adalah algoritma klasifikasi *Expectation Maximization* (EM), sebagaimana dijelaskan pada sub bab 2.8, metode ini membutuhkan sebuah model probabilitas awal, hasil dari model probabilitas *Naïve Bayes* dapat kita gunakan sebagai model probabilitas awal kemudian dilakukan tahap pertama dalam algoritma EM yaitu: tahap *Expectation* dan tahap *Maximization*.

3.2.4.1.2.1 *Expectation Step*

Expectation Step merupakan tahap memperkirakan kategori setiap dokumen yang tak berlabel dari matrik *term* dokumen berdasarkan dokumen yang berlabel, sehingga dibutuhkan nilai probabilitas awal, contoh berikut ini menggunakan kumpulan dokumen yang sama sebagaimana penentuan kategori sebuah dokumen menggunakan algoritma *Naïve Bayes* pada sub-bab 3.2.4.1.1 ditambah dengan satu buah dokumen 3 yang tak berlabel.

Tabel 3.6 Tabel Matrik Term Dokumen

| Dokumen | <i>Term</i> | | | | | | | | | | | | | | | |
|---------|-------------|------|-------|----------|-------|------|--------|----|---------|-------|--------------|--------|--------|---------|-----------|--------|
| | Achieve | Call | Check | District | Enter | Exam | Health | ID | Limited | Print | Registration | School | Select | Student | Unlimited | Verify |
| D1 | 4 | 4 | 2 | 0 | 0 | 1 | 1 | 0 | 0 | 2 | 4 | 0 | 0 | 4 | 0 | 2 |
| D2 | 0 | 0 | 0 | 1 | 3 | 0 | 0 | 3 | 1 | 3 | 14 | 3 | 3 | 3 | 1 | 3 |
| D3 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 5 | 1 | 1 | 2 | 0 | 1 |

Tabel 3.7 Tabel Nilai Probabilitas

| Kategori | $p(c_i)$ | $p(w_{kj}, c_i)$ | | | | | | | | | | | | | | | |
|------------------|----------|------------------|------|-------|----------|-------|------|--------|------|---------|-------|--------------|--------|--------|---------|-----------|--------|
| | | Achieve | Call | Check | District | Enter | Exam | Health | ID | Limited | Print | Registration | School | Select | Student | Unlimited | Verify |
| Pre Registration | 1/2 | 5/25 | 5/25 | 3/25 | 1/25 | 1/25 | 2/25 | 2/25 | 1/25 | 1/25 | 3/25 | 5/25 | 1/25 | 1/25 | 5/25 | 1/25 | 3/25 |
| Registration | 1/2 | 1/27 | 1/27 | 1/27 | 2/27 | 4/27 | 1/27 | 1/27 | 4/27 | 2/27 | 4/27 | 15/27 | 4/27 | 4/27 | 4/27 | 2/27 | 4/27 |

yang dapat dilihat pada Tabel 3.6, sehingga nilai probabilitas pada Tabel 3.7 dapat dijadikan sebagai nilai probabilitas awal, nilai probabilitas awal dihitung menggunakan Persamaan 2.7.

- $$\begin{aligned}
 p("pre"|"d3") &= (p("pre") \times p("verify"|"pre") \times \\
 &p("registration"|"pre") \times p("student"|"pre")) : ((p("pre") \times \\
 &p("verify"|"pre") \times p("registration"|"pre") \times p("student"|"pre")) + \\
 &(p("reg") \times p("verify"|"reg") \times p("registration"|"reg") \times \\
 &p("student"|"reg"))) \\
 &= (1/2 \times 3/25 \times 5/25 \times 5/25) : ((1/2 \times 3/25 \times 5/25 \times 5/25) + (1/2 \times \\
 &4/27 \times 15/27 \times 4/27)) \\
 &= \mathbf{0,282464877}
 \end{aligned}$$
- $$\begin{aligned}
 p("reg"|"d3") &= (p("reg") \times p("verify"|"reg") \times \\
 &p("registration"|"reg") \times p("student"|"reg")) : ((p("pre") \\
 &\times p("verify"|"pre") \times p("registration"|"pre") \times p("student"|"pre")) + \\
 &(p("reg") \times p("verify"|"reg") \times p("registration"|"reg") \\
 &\times p("student"|"reg"))) \\
 &= (1/2 \times 4/27 \times 15/27 \times 4/27) : ((1/2 \times 3/25 \times 5/25 \times 5/25) + (1/2 \times 4/27 \\
 &\times 15/27 \times 4/27)) \\
 &= \mathbf{0,717535123}
 \end{aligned}$$
- Karena $p("reg"|"d3") > p("pre"|"d3")$, maka kategori dari dokumen 3 adalah **Registration**.

3.2.4.1.2.2 Maximization Step

Tahap berikutnya adalah *Maximization Step* sebagaimana dijelaskan pada sub bab 2.8, tujuan dari *Maximization Step* memper barui nilai probabilitas awal, hingga didapatkan nilai maksimal yaitu 1, perhitungan *Maximization Step* didasarkan pada Persamaan 2.8, untuk mendapatkan nilai maksimal perlu kita minimalkan nilai probabilitas dari term, dimana pada matrik term dokumen masih terdapat nilai kemunculan dari term bernilai 0 sehingga perlu dilakukan perhitungan untuk nilai kemunculan keseluruhan term $f(p)$. Berikut contoh perhitungan $f(p)$ berdasarkan Tabel 3.6.

- $$\begin{aligned}
f(p) &= 16 + N(\text{"achieve", "d1"}) p(\text{"pre"}|\text{"d1"}) + N(\text{"call", "d1"}) \\
&p(\text{"pre"}|\text{"d1"}) + N(\text{"check", "d1"}) p(\text{"pre"}|\text{"d1"}) + N(\text{"exam", "d1"}) \\
&p(\text{"pre"}|\text{"d1"}) + N(\text{"health", "d1"}) p(\text{"pre"}|\text{"d1"}) + N(\text{"print", "d1"}) \\
&p(\text{"pre"}|\text{"d1"}) + N(\text{"registration", "d1"}) p(\text{"pre"}|\text{"d1"}) + N(\text{"student", \\
&\text{"d1"}}) p(\text{"pre"}|\text{"d1"}) + N(\text{"verify", "d1"}) p(\text{"pre"}|\text{"d1"}) + N(\text{"district", \\
&\text{"d2"}}) p(\text{"reg"}|\text{"d2"}) + N(\text{"enter", "d2"}) p(\text{"pre"}|\text{"d2"}) + N(\text{"enter", \\
&\text{"d3"}}) p(\text{"pre"}|\text{"d3"}) + N(\text{"enter", "d3"}) p(\text{"reg"}|\text{"d3"}) + N(\text{"ID", \\
&\text{"d2"}}) p(\text{"reg"}|\text{"d2"}) + N(\text{"ID", "d3"}) p(\text{"pre"}|\text{"d3"}) + N(\text{"ID", "d3"}) \\
&p(\text{"reg"}|\text{"d3"}) + N(\text{"limited", "d2"}) p(\text{"reg"}|\text{"d2"}) + N(\text{"print", "d2"}) \\
&p(\text{"reg"}|\text{"d2"}) + N(\text{"print", "d3"}) p(\text{"pre"}|\text{"d3"}) + N(\text{"print", "d3"}) \\
&p(\text{"reg"}|\text{"d3"}) + N(\text{"registration", "d2"}) p(\text{"reg"}|\text{"d2"}) + \\
&N(\text{"registration", "d3"}) p(\text{"pre"}|\text{"d2"}) + N(\text{"registration", "d3"}) \\
&p(\text{"reg"}|\text{"d3"}) + N(\text{"school", "d2"}) p(\text{"reg"}|\text{"d2"}) + N(\text{"school", "d3"}) \\
&p(\text{"pre"}|\text{"d3"}) + N(\text{"school", "d3"}) p(\text{"reg"}|\text{"d3"}) + N(\text{"select", "d2"}) \\
&p(\text{"reg"}|\text{"d2"}) + N(\text{"select", "d3"}) p(\text{"pre"}|\text{"d3"}) + N(\text{"select", "d3"}) \\
&p(\text{"reg"}|\text{"d3"}) + N(\text{"student", "d2"}) p(\text{"reg"}|\text{"d2"}) + N(\text{"student", \\
&\text{"d3"}}) p(\text{"pre"}|\text{"d3"}) + N(\text{"student", "d3"}) p(\text{"reg"}|\text{"d3"}) + \\
&N(\text{"unlimited", "d2"}) p(\text{"reg"}|\text{"d2"}) + N(\text{"verify", "d2"}) p(\text{"reg"}|\text{"d2"}) \\
&+ N(\text{"verify", "d3"}) p(\text{"pre"}|\text{"d3"}) + N(\text{"verify", "d3"}) p(\text{"reg"}|\text{"d3"}) \\
&= 16 + 4 \times 1 + 4 \times 1 + 2 \times 1 + 1 \times 1 + 1 \times 1 + 2 \times 1 + 4 \times 1 + 4 \times 1 \\
&+ 2 \times 1 + 1 \times 1 + 3 \times 1 + 1 \times 0,282464877 + 1 \times 0,717535123 + 3 \times 1 + 2 \\
&\times 0,282464877 + 2 \times 0,717535123 + 1 \times 1 + 3 \times 1 + 1 \times 0,282464877 + 1 \\
&\times 0,717535123 + 14 \times 1 + 5 \times 0,282464877 + 5 \times 0,717535123 + 3 \times 1 + \\
&1 \times 0,282464877 + 1 \times 0,717535123 + 3 \times 1 + 1 \times 0,282464877 + 1 \times \\
&0,717535123 + 3 \times 1 + 2 \times 0,282464877 + 2 \times 0,717535123 + 1 \times 1 + 3 \times \\
&1 + 1 \times 0,282464877 + 1 \times 0,717535123 \\
&= 92
\end{aligned}$$

Langkah berikutnya melakukan perhitungan probabilitas setiap term dengan kategori, berikut contoh perhitungan probabilitas term *achieve* dalam kategori *Pre Registration* dan *Registration*.

- $p(\text{"achieve"}|\text{"pre"}) = (1 + N(\text{"achieve", "d1"}) p(\text{"pre"}|\text{"d1"}) + N(\text{"achieve", "d2"}) p(\text{"pre"}|\text{"d2"}) + N(\text{"achieve", "d3"}) p(\text{"pre"}|\text{"d3"})) : f(p)$

$$= 1 + 4 \times 1 + 0 \times 0 + 0 \times 0,282464877 : 92$$

$$= \mathbf{0,054347826}$$

- $p(\text{"achieve"}|\text{"reg"}) = (1 + N(\text{"achieve", "d1"}) p(\text{"reg"}|\text{"d1"}) + N(\text{"achieve", "d2"}) p(\text{"reg"}|\text{"d2"}) + N(\text{"achieve", "d3"}) p(\text{"reg"}|\text{"d3"})) : f(p)$

$$= 1 + 4 \times 0 + 0 \times 1 + 0 \times 0,282464877 : 92$$

$$= \mathbf{0,010869565}$$

Langkah terakhir untuk menyelesaikan *Maximization Step* adalah memper barui nilai probabilitas untuk setiap kategori yang ada, dimana:

$|w|$: Jumlah semua kategori, maka bernilai 2

$|D|$: Jumlah seluruh data *training* dokumen, maka bernilai 3

- $p(\text{"pre"}) = (1 + p(\text{"pre"}|\text{"d1"}) + p(\text{"pre"}|\text{"d2"}) + p(\text{"pre"}|\text{"d3"})) : (2+3)$

$$= (1 + 1 + 0 + 0,282464877) / 5$$

$$= \mathbf{0,456492975}$$

- $p(\text{"reg"}) = (1 + p(\text{"reg"}|\text{"d1"}) + p(\text{"reg"}|\text{"d2"}) + p(\text{"reg"}|\text{"d3"})) : (2+3)$

$$= (1 + 0 + 1 + 0,717535123) / 5$$

$$= \mathbf{0,543507025}$$

Sehingga model probabilitas setelah *Maximization Step* di tunjukan pada Tabel 3.8.

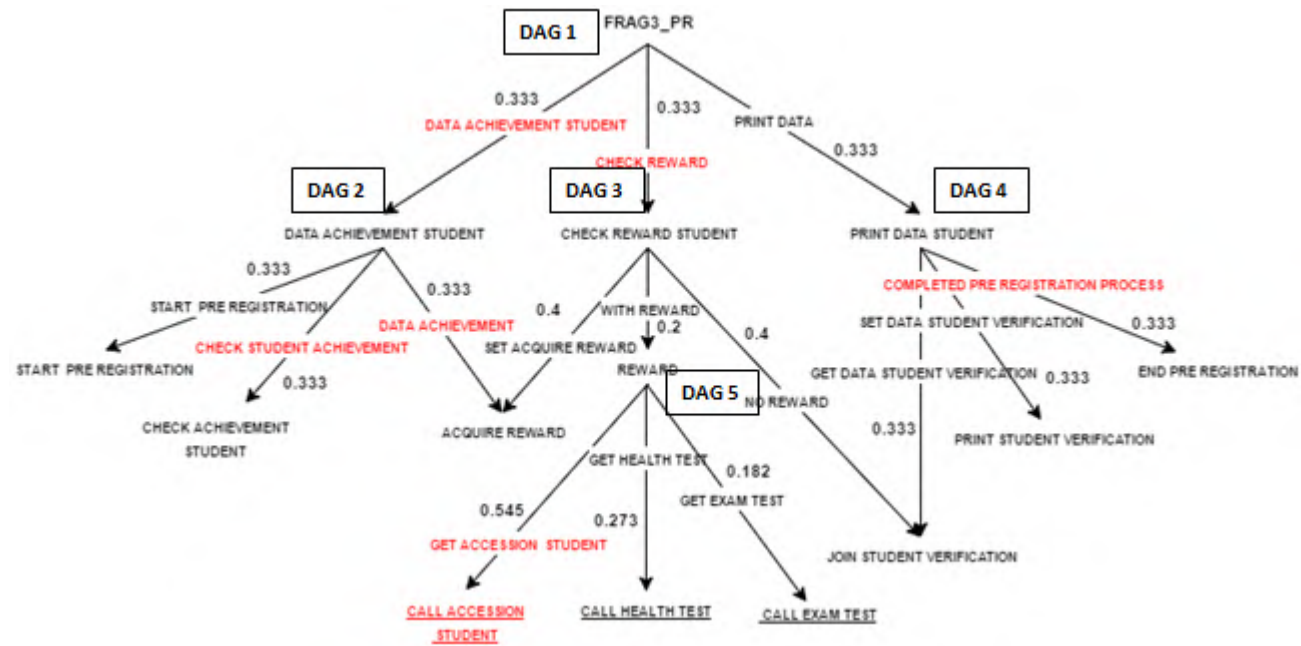
Tabel 3.8 Tabel Nilai Probabilitas Setelah *Maximization Step*

| Kategori | $p(c_i)$ | $p(w_{kj}, c_i)$ | | | | | | | | | | | | | | | |
|------------------|----------|------------------|--------|--------|----------|--------|--------|--------|--------|---------|--------|--------------|--------|--------|---------|-----------|--------|
| | | Achieve | Call | Check | District | Enter | Exam | Health | ID | Limited | Print | Registration | School | Select | Student | Unlimited | Verify |
| Pre Registration | 0,456 | 0,0543 | 0,0543 | 0,0326 | 0,0109 | 0,0139 | 0,0217 | 0,0217 | 0,0139 | 0,0109 | 0,0697 | 0,0139 | 0,0139 | 0,0139 | 0,0605 | 0,0109 | 0,0357 |
| Registration | 0,544 | 0,0109 | 0,0543 | 0,0109 | 0,0217 | 0,0404 | 0,0109 | 0,0109 | 0,0513 | 0,2020 | 0,0513 | 0,0404 | 0,0513 | 0,0513 | 0,0591 | 0,0217 | 0,0513 |

Probabilitas total dari kategori yang ada dari Tabel 3.8 adalah 1, hal ini menunjukkan bahwa nilai model probabilitas telah maksimal, apabila nilai model probabilitas belum maksimal maka proses *Expectation* dan *Maximization Step* dilakukan beberapa kali iterasi, setelah model probabilitas telah didapat maka kita dapat menghitung penentuan kategori dari sebuah dokumen, berikut contoh perhitungan penentuan kategori dari dokumen 3 berdasarkan Persamaan 2.6.

- $$\begin{aligned}
 p("pre"|"d3") &= p("pre") \times p("print"|"pre") \times p("registration"|"pre") \\
 &\quad \times p("student"|"pre") \\
 &= 0,45649 \times 0,06970 \times 0,01394 \times 0,06049 \\
 &= \mathbf{0,0000268}
 \end{aligned}$$
- $$\begin{aligned}
 p("reg"|"d3") &= p("reg") \times p("print"|"reg") \times p("registration"|"reg") \\
 &\quad \times p("student"|"reg") \\
 &= 0,54351 \times 0,05128 \times 0,04041 \times 0,05908 \\
 &= \mathbf{0,0000665}
 \end{aligned}$$

Sehingga dapat ditarik kesimpulan karena karena $p("reg"|"d3") > p("pre"|"d3")$, maka kategori dari **dokumen 3** adalah **Registration**.



Gambar 3.11 Representasi Model *Pre Registrasi 3* Dalam *Weighted Directed Acyclic Graph (WDAG)*

3.2.5. Perhitungan Kesamaan WDAG

Pada tahap ini akan dilakukan perhitungan kesamaan WDAG, Pada Gambar 3.11 merupakan representasi model proses dalam WDAG dengan label dan bobot dari setiap *edge*. Perhitungan kesamaan diantara model proses menggunakan kesamaan WDAG ini juga akan memperhatikan pencocokan string secara semantik, sehingga diharapkan dapat memberikan hasil perhitungan kesamaan yang lebih baik. Untuk melakukan perhitungan kesamaan WDAG kami menggunakan metode yang dilakukan oleh (Jin, 2006), sebagaimana telah dijelaskan pada sub-bab 2.10.2, perhitungan kesamaan WDAG dilakukan menggunakan Persamaan 2.11 dan Persamaan 2.12.

3.3 Pengujian dan Hasil Analisis

Skenario uji coba sistem atas metode yang disampaikan dilakukan dengan skenario membandingkan dan menghitung nilai kesamaan WDAG dari model *fragment* yang sama dan nilai kesamaan WDAG dari model *fragment* yang berbeda, kemudian hasil dari pengujian dianalisis menggunakan *Confusion Matrix* yang merupakan metode evaluasi model klasifikasi untuk memperkirakan objek yang benar atau salah, sebuah matriks dari prediksi yang akan dibandingkan dengan kelas yang asli dari inputan atau dengan kata lain berisi informasi nilai aktual dan prediksi pada klasifikasi.

Tabel 3.9 Tabel Confusion Matrix

| Klasifikasi | Prediksi Kelas | |
|-------------|----------------|-------------|
| | Kelas True | Kelas False |
| Kelas True | A = TP | B = FN |
| Kelas False | C = FP | D = TN |

Rumus untuk menghitung tingkat akurasi berdasarkan Tabel 3.9 menggunakan Persamaan 3.3:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} = \frac{A+D}{A+C+B+D} \quad (3.3)$$

Data pengujian diolah untuk mendapatkan tingkat akurasi hasil prediksi berupa jumlah *True Positive (TP)*, *True Negative (TN)*, *False Positive (FP)*, dan *False Negative (FN)* seperti yang dapat di lihat pada Tabel 3.9, hasil positif merujuk pada model proses yang diprediksikan masuk ke dalam kategori sebenarnya dan hasil negatif merujuk pada model proses yang diprediksikan masuk ke dalam kategori salah.

3.4 Pembuatan Perangkat Lunak

Pada tahap ini akan dibangun sebuah perangkat lunak berbasis desktop menggunakan bahasa pemrograman Java. Perangkat lunak yang akan dibangun akan menerima masukan berupa bisnis proses dengan mendefinisikan node dan edge, kemudian tahap berikutnya adalah memecah menjadi model proses dan kemudian dianalisis menggunakan metode struktural dan semantik, output/hasil dari perangkat lunak ini menghasilkan nilai kesamaan dari setiap model proses.

Lingkungan yang dilakukan dalam penelitian ini adalah sebagai berikut:

1. Satu buah PC dengan *Memory* yang memiliki spesifikasi *Processor* Intel *Core i5-M430 CPU @ 2.13GHz*, memori RAM 2 GB, dan sistem operasi Microsoft Windows 10 Enterprise 64 bit.
2. MySQL 5.0
3. Python 2.7.11
4. Netbeans 8.0

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil Penelitian

Penelitian ini diimplementasikan kedalam beberapa perangkat lunak sebagai sistem pendukung yang berfungsi untuk:

- Mendekomposisi/menguraikan model proses bisnis keseluruhan yang dimodelkan menggunakan *Workflow* menggunakan program Java.
- Mentransformasikan hasil dekomposisi proses bisnis kedalam *Weighted Directed Acyclic Graph* (WDAG) menggunakan *Triconnected Abstraction*.
- Menentukan bobot WDAG dari *fragment* dengan menggunakan *Process Vector*.
- Menentukan nilai probabilitas dari term kedalam beberapa topik yang akan digunakan sebagai analisa *tekstual* pada label *Node* dan *Edge* dari WDAG.
- Menghitung tingkat kesamaan (*similarity*) diantara model dari proses bisnis menggunakan WDAG *Similarity*.

Hasil implementasi dan pengujian dalam sistem akan dijelaskan secara lebih detail pada bagian berikut.

4.1.1 Komposisi dalam Proses Bisnis PPDB

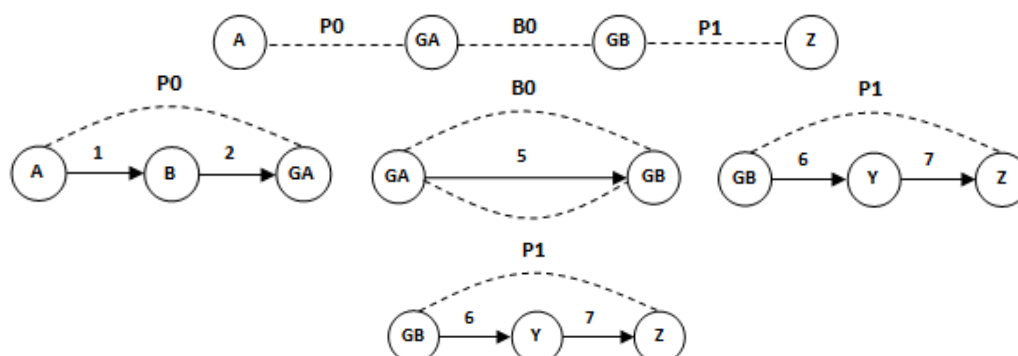
Proses bisnis PPDB yang menjadi studi kasus dalam penelitian ini mencakup tiga proses utama yang masing-masing memiliki variasi proses bisnis, dalam penerapannya, variasi proses ini melibatkan sub-proses yang dapat dijalankan pula oleh variasi yang lain dalam kelompok proses yang sama. Jumlah variasi dan sub-proses dari setiap kelompok proses, dapat dijelaskan berikut ini.

1. Proses bisnis Pendataan Siswa (*Student Collection*), dengan 4 model utama proses dan memiliki 2 model variasi yaitu: proses bisnis pendataan nilai siswa, dan verifikasi siswa domisili dalam kota/kabupaten.

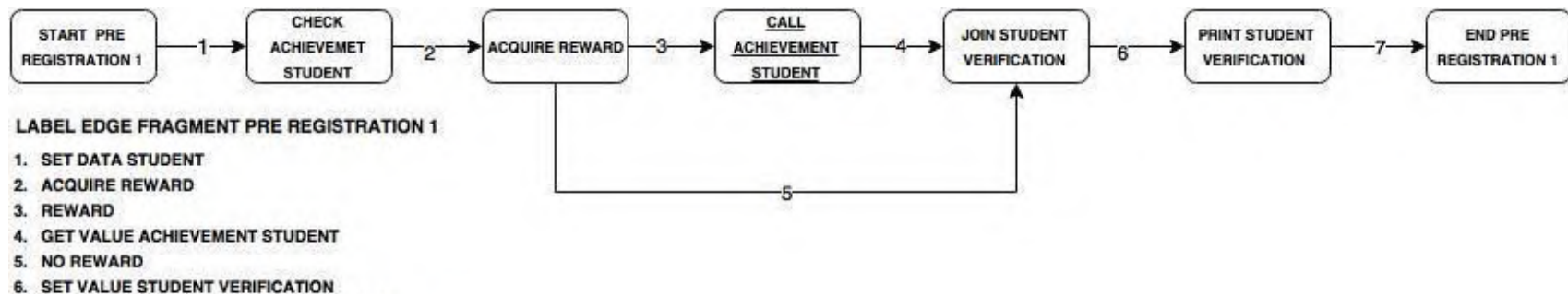
2. Proses bisnis Pra Pendaftaran (*Pre Registration*), dengan 5 model utama proses dan memiliki 5 model variasi yaitu: proses bisnis penilaian tes kesehatan, penilaian tes tambahan, penilaian tes ujian masuk, penilaian berbasis prestasi, dan pembobotan nilai berdasarkan domisili siswa.
3. Proses bisnis Pendaftaran (*Registration*), dengan 4 model utama proses dan memiliki 4 model variasi yaitu: proses bisnis untuk pemilihan sekolah dengan jumlah tidak terbatas, pemilihan sekolah dengan jumlah terbatas, pemilihan sekolah berdasarkan rayonisasi, dan pemilihan sekolah kejuruan.

4.1.2 Hasil Dekomposisi Proses Bisnis menggunakan RPST dan SESE

Dekomposisi proses bisnis digunakan untuk memecah model proses agar lebih mudah dianalisa dapat diketahui sub-proses yang membangun proses tertentu, pada penelitian ini total model proses bisnis yang didekomposisi berjumlah 24 model proses bisnis, keterlibatan sub proses dalam setiap model proses bisnis turut mempengaruhi jumlah *fragment* dari hasil dekomposisi menggunakan RPST. Tabel 4.1 menunjukkan sub-proses yang membentuk sebuah *fragment* dari model utama/variasi model dan dapat dilihat bahwa beberapa sub-proses dari *fragment* memiliki kesamaan sub-proses dari *fragment* lainnya, sedangkan Tabel 4.2 merupakan jumlah *fragment* hasil dekomposisi RPST dan SESE dari model utama dan model variasi, hasil dekomposisi dapat dilihat pada Gambar 4.1.



Gambar 4.1 Gambar Hasil Dekomposisi Model *Pre Registration* 1



Gambar 4.1. Gambar *Fragment Model*

Tabel 4.1.a Sub-Proses *Fragment*

| NAMA FRAGMENT | DETAIL SUB PROSES PRE REGISTRATION | MODEL |
|-------------------|---|------------------|
| FRAGMENT1_PR | START PRE REGISTRATION, CHECK ACHIEVEMENT STUDENT, ACQUIRE REWARD | MODEL_PR1 |
| FRAGMENT2_PR | ACQUIRE REWARD, CALL ACHIEVEMENT STUDENT, JOIN STUDENT VERIFICATION | MODEL_PR1 |
| FRAGMENT3_PR | JOIN STUDENT VERIFICATION, PRINT STUDENT VERIFICATION, END PRE REGISTRATION | MODEL_PR1 |
| ... | ... | ... |
| ... | ... | ... |
| FRAGMENT15_PR | JOIN STUDENT VERIFICATION, PRINT STUDENT VERIFICATION, END PRE REGISTRATION | MODEL_PR5 |
| NAMA FRAGMENT | DETAIL SUB PROSES PRE REGISTRATION | MODEL |
| FRAGMENT1_VAR_PR | START ACHIEVEMENT SCORE, ENTER FIELD ACHIEVEMENT, CHOOSE ACHIEVEMENT | MODEL_ACV_REG_PR |
| FRAGMENT2_VAR_PR | SET ACHIEVEMENT, ENTER SPORT ACHIEVEMENT, JOIN ACHIEVEMENT SCORE | MODEL_ACV_REG_PR |
| FRAGMENT3_VAR_PR | SET ACHIEVEMENT, ENTER ART ACHIEVEMENT, JOIN ACHIEVEMENT SCORE | MODEL_ACV_REG_PR |
| ... | ... | ... |
| ... | ... | ... |
| FRAGMENT23_VAR_PR | JOIN HEALTH SCORE, SET HEALTH SCORE, END HEALTH TEST | MODEL_HLT_REG_PR |
| NAMA FRAGMENT | DETAIL SUB PROSES PRE REGISTRATION | MODEL |
| FRAGMENT1_REG | START REGISTRATION, SELECT DISTRICT SCHOOL, PRINT REGISTRATION, | MODEL_REG1 |
| | ENTER ID REGISTRATION, VERIFY REGISTRATION, END REGISTRATION | |
| ... | ... | ... |

Tabel 4.1.b. Sub-Proses *Fragment Lanjutan*

| | | |
|----------------------|---|-------------------|
| ... | ... | ... |
| FRAGMENT4_REG | START REGISTRATION, SELECT VOCATIONAL SCHOOL, PRINT REGISTRATION, | MODEL_REG4 |
| | ENTER ID REGISTRATION, VERIFY REGISTRATION, END REGISTRATION | |
| NAMA FRAGMENT | DETAIL SUB PROSES PRE REGISTRATION | MODEL |
| FRAGMENT1_PR | START PRE REGISTRATION, CHECK ACHIEVEMENT STUDENT, ACQUIRE REWARD | MODEL_PR1 |
| FRAGMENT2_PR | ACQUIRE REWARD, CALL ACHIEVEMENT STUDENT, JOIN STUDENT VERIFICATION | MODEL_PR1 |
| FRAGMENT3_PR | JOIN STUDENT VERIFICATION, PRINT STUDENT VERIFICATION, END PRE REGISTRATION | MODEL_PR1 |
| ... | ... | ... |
| ... | ... | ... |
| FRAGMENT15_PR | JOIN STUDENT VERIFICATION, PRINT STUDENT VERIFICATION, END PRE REGISTRATION | MODEL_PR5 |
| FRAGMENT1_PR | START PRE REGISTRATION, CHECK ACHIEVEMENT STUDENT, ACQUIRE REWARD | MODEL_PR1 |
| NAMA FRAGMENT | DETAIL SUB PROSES STUDENT COLLECTION | MODEL ASAL |
| FRAGMENT1_SC | START STUDENT COLLECTION, ENTER STUDENT COLLECTION, JOIN BIODATA INPUT | MODEL_SC1 |
| FRAGMENT2_SC | JOIN BIODATA INPUT, ENTER ID NUMBER, ENTER STUDENT ORIGIN, CHECK LOCAL STUDENT | MODEL_SC1 |
| FRAGMENT3_SC | CHECK LOCAL STUDENT, ENTER BIODATA, ENTER GRADE, JOINT ADDITIONAL INPUT | MODEL_SC1 |
| ... | ... | ... |
| ... | ... | ... |
| FRAGMENT 17_SC | JOIN BIODATA INPUT, ENTER ID NUMBER, CHECK ID NUMBER, CHECK VALID STUDENT | MODEL_SC4 |
| | CHECK VALID STUDENT, ENTER BIODATA, ENTER GRADE, ENTER SUPPLEMENTARY DATA, VERIFY STUDENT | |
| | COLLECTION, END STUDENT COLLECTION | |
| NAMA FRAGMENT | DETAIL SUB PROSES STUDENT COLLECTION | MODEL ASAL |
| FRAGMENT 1_VAR_SC | START DOMICILE VERIFICATION, ENTER NIK, CHECK NATIONAL DATABASE, CHECK VALID CITIZEN | MODEL_DOM_VER_SC |
| FRAGMENT 2_VAR_SC | CHECK VALID CITIZEN, EXTRACT FAMILY DATABASE, STUDENT DATA EXCHANGING | MODEL_DOM_VER_SC |
| FRAGMENT 3_VAR_SC | CHECK VALID CITIZEN, REGISTER NIK, VALID CITIZEN | MODEL _VER_SC |
| ... | ... | ... |

Tabel 4.2 Jumlah *Fragment*

| NO | NAMA MODEL | JML FRAGMENT | JENIS |
|-------|-------------------|--------------|---------------|
| 1. | MODEL_PR1 | 3 | MODEL UTAMA |
| 2. | MODEL_PR2 | 3 | MODEL UTAMA |
| 3. | MODEL_PR3 | 3 | MODEL UTAMA |
| 4. | MODEL_PR4 | 3 | MODEL UTAMA |
| 5. | MODEL_PR5 | 3 | MODEL UTAMA |
| 6. | MODEL_ACV_REG_PR | 5 | MODEL VARIASI |
| 7. | MODEL_ADS_REG_PR | 4 | MODEL VARIASI |
| 8. | MODEL_DOM_REG_PR | 5 | MODEL VARIASI |
| 9. | MODEL_EXM_REG_PR | 4 | MODEL VARIASI |
| 10. | MODEL_HLT_REG_PR | 7 | MODEL VARIASI |
| 11. | MODEL_REG1 | 1 | MODEL UTAMA |
| 12. | MODEL_REG2 | 1 | MODEL UTAMA |
| 13. | MODEL_REG3 | 1 | MODEL UTAMA |
| 14. | MODEL_REG4 | 1 | MODEL UTAMA |
| 15. | MODEL_DIS_SCH_REG | 3 | MODEL VARIASI |
| 16. | MODEL_VOC_SCH_REG | 5 | MODEL VARIASI |
| 17. | MODEL_LIM_SCH_REG | 4 | MODEL VARIASI |
| 18. | MODEL_UNL_SCH_REG | 3 | MODEL VARIASI |
| 19. | MODEL_SC1 | 6 | MODEL UTAMA |
| 20. | MODEL_SC2 | 5 | MODEL UTAMA |
| 21. | MODEL_SC3 | 5 | MODEL UTAMA |
| 22. | MODEL_SC4 | 3 | MODEL UTAMA |
| 23. | MODEL_DOM_VER_SC | 4 | MODEL VARIASI |
| 24. | MODEL_ENT_SG_SC | 5 | MODEL VARIASI |
| TOTAL | | 87 | |

4.1.3 Hasil Pembobotan *Edge* WDAG Menggunakan WCDG

Hasil pembobotan *edge* dari sebuah WDAG, menggunakan pendekatan *Weighted Completed Dependency Graph* (WCDG), WCDG digunakan untuk mengetahui *Dependency activity* dan *transition* sebuah proses berdasarkan kompleksitas yang mencerminkan interaksi di antara node sehingga dapat diukur dengan menggunakan pendekatan *Control Flow Complexity* sebagai nilai probabilitas dalam percabangan. Pembobotan ini dilakukan pada setiap sub-proses yang membentuk *fragment*, contoh pembahasan pada Tabel 4.1.a terdapat *fragment* model “FRAGMENT1_PR” yang merupakan *fragment* model dari MODEL_PR1 dimana *fragment* model terdiri dari beberapa sub proses, sehingga pembobotan sub-proses dapat dilakukan sebagai representasi bobot model *fragment* untuk dijadikan sebagai bobot dari WDAG. Tabel 4.3 merupakan contoh

perhitungan bobot untuk representasi WDAG. Gambar 4.1 merupakan *fragment* model “FRAGMENT1_PR”.

Tabel 4.3 Tabel Proses Pembobotan “FRAGMENT1_PR”

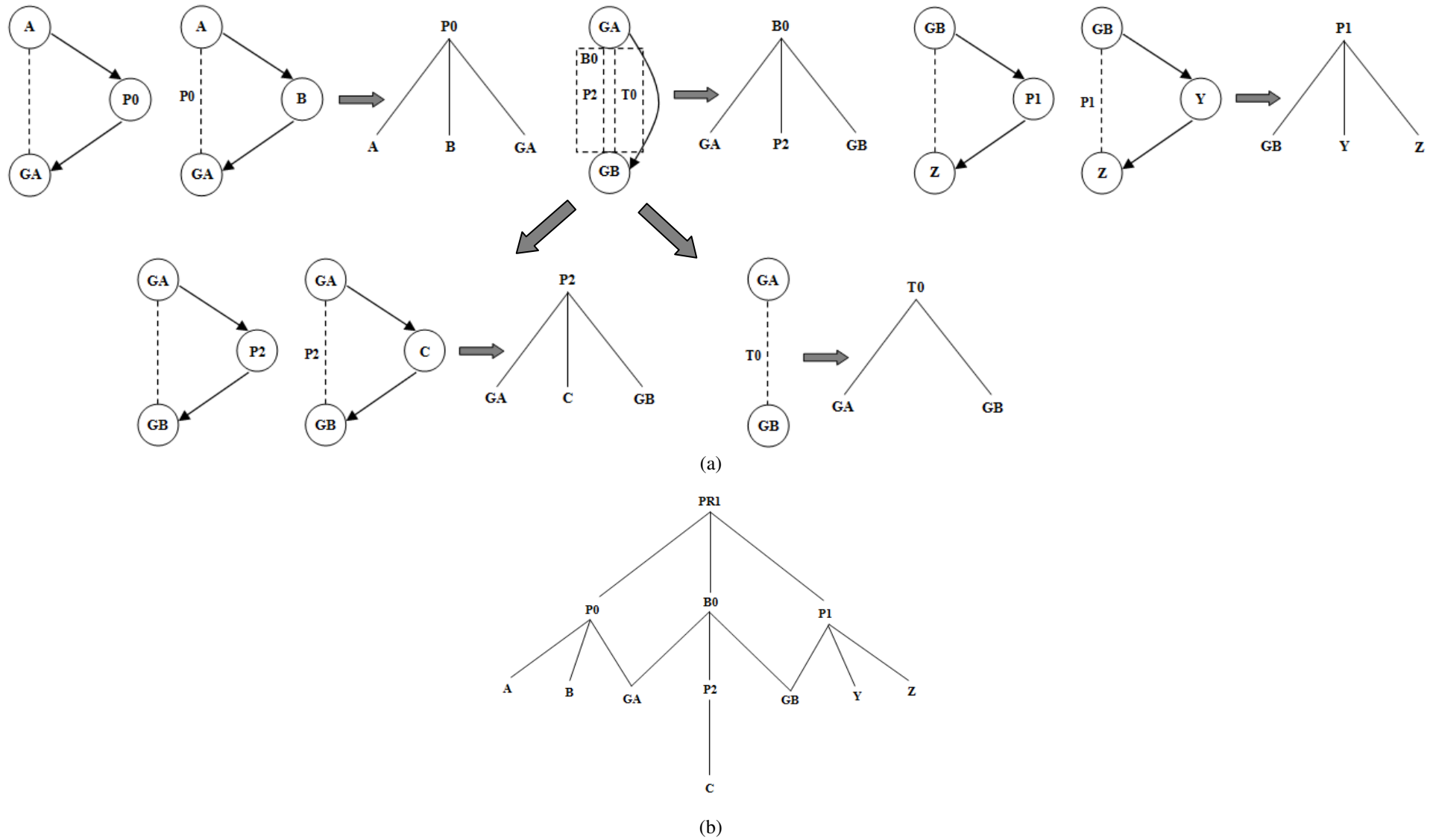
| PROSES1_PR | PEMBOBOTAN | | | |
|---------------------------|------------|----------|-------|------------|
| NAMA SUB PROSES | VALUE NODE | DISTANCE | BOBOT | BOBOT WDAG |
| START PRE REGISTRATION | 1 | | 1 | 0,33333 |
| CHECK ACHIEVEMENT STUDENT | 1 | 1 | 1 | 0,33333 |
| ACQUIRE REWARD | 1 | | 1 | 0,33333 |
| | | | 3 | 1 |

| PROSES2_PR | PEMBOBOTAN | | | |
|---------------------------|------------|----------|-------|------------|
| NAMA SUB PROSES | VALUE NODE | DISTANCE | BOBOT | BOBOT WDAG |
| ACQUIRE REWARD | 1 | | 1 | 0,4 |
| CALL ACHIEVEMENT STUDENT | 0,5 | 1 | 0,5 | 0,2 |
| JOIN STUDENT VERIFICATION | 1 | | 1 | 0,4 |
| | | | 2,5 | 1 |

| PROSES3_PR | PEMBOBOTAN | | | |
|----------------------------|------------|----------|-------|------------|
| NAMA SUB PROSES | VALUE NODE | DISTANCE | BOBOT | BOBOT WDAG |
| JOIN STUDENT VERIFICATION | 1 | | 1 | 0,33333 |
| PRINT STUDENT VERIFICATION | 1 | 1 | 1 | 0,33333 |
| END PRE REGISTRATION | 1 | | 1 | 0,33333 |
| | | | 3 | 1 |

4.1.4 Hasil Representasi WDAG Menggunakan *Triconnected Abstraction*

Representasi WDAG pada penelitian ini menggunakan metode *Triconnected Abstraction* (*Trivial Abstraction*, *Polygon Abstraction*, *Bond Abstraction*, dan *Rigid Abstraction*), data yang digunakan untuk representasi ini menggunakan data hasil dekomposisi dari Tabel 4.1, sebagai contoh ditampilkan pada Gambar 4.2 *fragment* model *Pre Registration* 1 direpresentasikan menggunakan *Polygon Abstraction* dan *Bond Abstraction*.



Gambar 4.2 *Triconnected Abstraction* (a) *Fragment Pre Registration 1*, (b) *WDAG Pre Registration 1*

4.1.5 Hasil Pemodelan Topik Menggunakan PLSA

Pemodelan topik menggunakan PLSA bertujuan untuk merepresentasikan kata-kata/*term* ke dalam 3 buah kategori sesuai komposisi proses bisnis PPDB, yaitu: *Pre Registration*, *Registration*, dan *Student Collection*, untuk mendapatkan kamus kata, penelitian ini menggunakan label aktivitas dan transisi dari sub-proses dengan tujuan mencari nilai kesamaan dari kata/*term* berdasarkan kategori komposisi proses bisnis PPDB, setelah dilakukan *pre-processing* dengan melakukan proses *tokenizing* didapatkan, didapatkan nilai probabilitas awal diantara kata/*term* menggunakan algoritma klasifikasi *Naïve Bayes*, pada Tabel 4.4. dan Tabel 4.5 merupakan dokumen yang digunakan sebagai percobaan untuk mendapatkan nilai probabilitas awal kategori dari tiap dokumen menggunakan algoritma *Naïve Bayes*, selanjutnya setelah didapatkan nilai probabilitas awal menggunakan algoritma *Naïve Bayes*, tahap berikutnya mencari nilai probabilitas yang optimal dari tabel probabilitas awal pada Tabel 4.4 menggunakan *Naïve Bayes*

Tabel 4.4 Tabel Probabilitas Awal Term Terhadap Topik

| NO | TERM | PROBABILITAS | | |
|-----|-----------|------------------|--------------|--------------------|
| | | PRE REGISTRATION | REGISTRATION | STUDENT COLLECTION |
| 1 | LIMITED | 0,363636225 | 0,381818418 | 0,254545357 |
| 2 | SELECTION | 0,470587986 | 0,49411788 | 0,035294134 |
| 3 | SHOW | 0,434782495 | 0,260869758 | 0,304347747 |
| 4 | ACCESSION | 0,649253959 | 0,141790953 | 0,208955088 |
| 5 | CALCULATE | 0,563380186 | 0,267605589 | 0,169014225 |
| ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... |
| 93 | VERIFY | 0,689654958 | 0,206896694 | 0,103448347 |
| 94 | ENTRY | 0,563380186 | 0,267605589 | 0,169014225 |
| 95 | REWARD | 0,671328892 | 0,132867044 | 0,195804064 |

Tabel 4.5 Tabel Probabilitas Awal Dokumen Terhadap Topik

| DOKUMEN | PROBABILITAS | | |
|---------|------------------|--------------|-----------------|
| | PRE REGISTRATION | REGISTRATION | STUDENT COLLECT |
| D11 | 0,3134554 | 0,151106153 | 0,202105114 |
| D41 | 0,195692615 | 0,224212533 | 0,168330146 |
| D60 | 0,096734878 | 0,043261907 | 0,164351041 |

algoritma *Expectation* dan *Maximization step*, Pada Tabel 4.6 merupakan nilai probabilitas yang dihasilkan menggunakan algoritma *EM step*. Tabel 4.7 merupakan dokumen yang digunakan sebagai percobaan dalam melakukan prediksi sebuah dokumen termasuk dalam kategori sebuah topik dengan menggunakan nilai probabilitas term terhadap topik dari Tabel 4.6, perhitungan nilai probabilitas dokumen telah optimal yaitu 1 pada tiap kategori topik, dimana dokumen D11 merupakan label *string* dari *fragment* model *pre registration* 4, dokumen D41 merupakan label *string* dari *fragment* model *registration* 1, dan dokumen D60 merupakan label *string* dari *fragment* model *student collection* 1.

Tabel 4.6. Tabel Probabilitas Term Terhadap Topik - EM Step

| NO | TERM | PROBABILITAS | | |
|-----|-----------|------------------|--------------|--------------------|
| | | PRE REGISTRATION | REGISTRATION | STUDENT COLLECTION |
| 1 | LIMITED | 0 | 0,03482587 | 0 |
| 2 | SELECTION | 0 | 0,03482587 | 0,00189394 |
| 3 | SHOW | 0 | 0,0199005 | 0 |
| 4 | ACCESSION | 0,03722721 | 0 | 0 |
| 5 | CALCULATE | 0 | 0 | 0,00757576 |
| ... | ... | ... | ... | ... |
| ... | ... | ... | ... | ... |
| 93 | VERIFY | 0 | 0,00995025 | 0,00378788 |
| 94 | ENTRY | 0 | 0 | 0,00757576 |
| 95 | REWARD | 0,04107831 | 0 | 0 |

Tabel 4.7. Tabel Probabilitas Dokumen Terhadap Topik - EM Step

| DOKUMEN | PROBABILITAS | | |
|---------|------------------|--------------|-----------------|
| | PRE REGISTRATION | REGISTRATION | STUDENT COLLECT |
| D11 | 1 | 0 | 0 |
| D41 | 0 | 1 | 0 |
| D60 | 0 | 0 | 1 |

4.2 Pembahasan Hasil Pengujian

Metode PLSA memungkinkan untuk mengetahui kesamaan term melalui pendekatan probabilitas term terhadap topik dari setiap label *Edge* yang dibandingkan, semakin tinggi nilai probabilitas sebuah term terhadap topik atau kategori maka semakin tinggi pula tingkat similaritas makna dari label *string*. Representasi pemodelan analisa gabungan menggunakan *Weighted Directed Acyclic Graph* (WDAG), dimana nilai kemiripan dihitung dengan menggunakan nilai *WDAGSimp* dan ketidakmiripan dari sebuah struktur dihitung berdasarkan nilai *WDAGplicity*.

4.2.1 Perbandingan Perhitungan WDAG Similarity

Skenario uji coba dilakukan, (1), skenario uji coba menggunakan perhitungan similaritas WDAG *Pure*, (2), skenario uji coba menggunakan metode yang diajukan, menggunakan perhitungan similaritas WDAG dimana nilai similaritas *tekstual* label *string* dihitung menggunakan *Cosine Similarity* dari *Probabilistic Latent Semantic Analysis* (PLSA), contoh hasil skenario uji coba pada model *fragment Pre Registration* ditunjukkan pada Tabel 4.8 .

Tabel 4.8. Contoh Hasil Uji Coba Perhitungan WDAG Similarity

| NO | WDAG 1 | WDAG 2 | WDAG PURE | WDAG + PLSA |
|----|--------|--------|-----------|-------------|
| 1 | PR1 | PR2 | 0,6043 | 0,783 |
| 2 | PR1 | PR3 | 0,6038 | 0,778 |
| 3 | PR2 | PR3 | 0,3423 | 0,7716 |
| 4 | PR2 | PR4 | 0,3434 | 0,7688 |
| 5 | PR2 | PR5 | 0,3434 | 0,7670 |
| 6 | PR3 | PR4 | 0,415 | 0,7835 |
| 7 | PR3 | PR5 | 0,416 | 0,7864 |
| 8 | PR4 | PR5 | 0,148 | 0,7346 |

Kami mengevaluasi kinerja sistem menggunakan *Confusion Matrix* untuk menguji tingkat akurasi dari hasil metode analisa gabungan, pada pengujian ini kami memakai 40 data uji, pada Tabel 4.9 menunjukkan matriks perbandingan yang menunjukkan evaluasi kinerja sistem, didapatkan nilai akurasi menggunakan metode WDAG *Pure* sebesar 0,05% dan menggunakan metode WDAG PLSA sebesar 45%.

Tabel 4.9 Matriks Perbandingan Untuk Pengukuran Kinerja Sistem

| WDAG <i>Pure</i> | | |
|------------------|-----------|-------|
| Prediksi | Kenyataan | Total |
| TRUE | TRUE | 2 |
| | FALSE | 18 |
| True Total | | 20 |
| FALSE | TRUE | 0 |
| | FALSE | 20 |
| False Total | | 20 |
| Jumlah Total | | 40 |

| WDAG PLSA | | |
|--------------|-----------|-------|
| Prediksi | Kenyataan | Total |
| TRUE | TRUE | 18 |
| | FALSE | 2 |
| True Total | | 20 |
| FALSE | TRUE | 0 |
| | FALSE | 20 |
| False Total | | 20 |
| Jumlah Total | | 40 |

4.2.2 Perbandingan Metode *Tekstual Similarity*

Untuk memeriksa kesamaan makna label *string* pada WDAG kami menggunakan PLSA dimana merupakan metode pemodelan topik pada dokumen, Hasil pemeriksaan kesamaan makna label *string* mempengaruhi hasil analisa kesamaan gabungan yang kami ajukan, pada penelitian ini kami membandingkan beberapa metode untuk pemeriksaan label *string* menggunakan data-set label *string* proses bisnis PPDB, diantaranya *Cosine Similarity Vector Space Model* (VSM), *Cosine Similarity Latent Semantic Analysis* (LSA) dan metode yang kami

gunakan yaitu *Cosine similarity Probabilistic Latent Semantic Analysis* (PLSA), hasil perbandingan dapat dilihat pada Tabel 4.10 dari ketiga metode yang dibandingkan kami mendapatkan nilai akurasi sebesar *Cosine Similarity VSM* (54,8%), *Cosine Similarity LSA* (50%) dan *Cosine Similarity PLSA* (59,5%).

Tabel 4.10 Tabel Contoh Hasil Perbandingan *Textual Similarity*

| LABEL STRING | | VSM | LSA | PLSA |
|----------------------------|-----------------------------|---------|---------|---------|
| DATA ACHIEVEMENT STUDENT | DATA ACCESSION STUDENT | 0,44325 | 0,66667 | 0,80318 |
| INPUT EXAM HEALTH | ENTRY TEST HEALTH | 0,166 | 0,33333 | 0,63323 |
| GET ACHIEVEMENT STUDENT | GET ACCESSION STUDENT | 0,44325 | 0,66667 | 0,80318 |
| CALL ACHIEVEMENT STUDENT | CALL ACCESSION STUDENT | 0,44325 | 0,66667 | 0,84281 |
| CHECK HEALTH | VERIFY HEALTH | 0,28473 | 0,5 | 0,932 |
| CHECK EXAM | VERIFY EXAM | 0,28473 | 0,5 | 0,42993 |
| SET DOMICILE REGISTRATION | SET RESIDENCE REGISTRATION | 0,166 | 0,66667 | 0,34222 |
| CALL DOMICILE REGISTRATION | CALL RESIDENCE REGISTRATION | 0,44325 | 0,33333 | 0,94693 |

4.2.3 Pengaruh *Process Vector* Terhadap Perhitungan *WDAG Similarity*

Process Vector pada penelitian ini digunakan sebagai bobot dalam *WDAG*, dalam *Process Vector*, *Dependency* menggambarkan hubungan interaksi di antara *Node*, nilai *Dependency* ini dipengaruhi oleh beberapa hal, pertama *Distance*/jarak dengan *Node Entry*, dimana semakin jauh sebuah *Node* dari *Node Entry* maka akan semakin kecil hubungan interaksinya, kedua, *Execution Probability* pada *Node Entry*, nilai *Execution Probability* menggambarkan tingkat kompleksitas aliran data pada percabangan.

4.2.4 Pengaruh *Tekstual Similarity* Terhadap *WDAG Similarity*

Pada perhitungan *WDAG Similarity* umumnya dibutuhkan pengecekan kesamaan makna label *string* pada *Edge* dan *Node*, pengecekan ini menghasilkan nilai 0 dan 1, memperoleh nilai 0 ketika label *string* yang diuji berbeda dan memperoleh nilai 1 pada saat label *string* yang diuji sama, dengan menggunakan metode *Tekstual Similarity* dimungkinkan untuk melakukan pengecekan pada label *string* yang memiliki persamaan arti (sinonim) sehingga dapat membantu meningkatkan kinerja perhitungan *WDAG Similarity* dimana label *string* yang

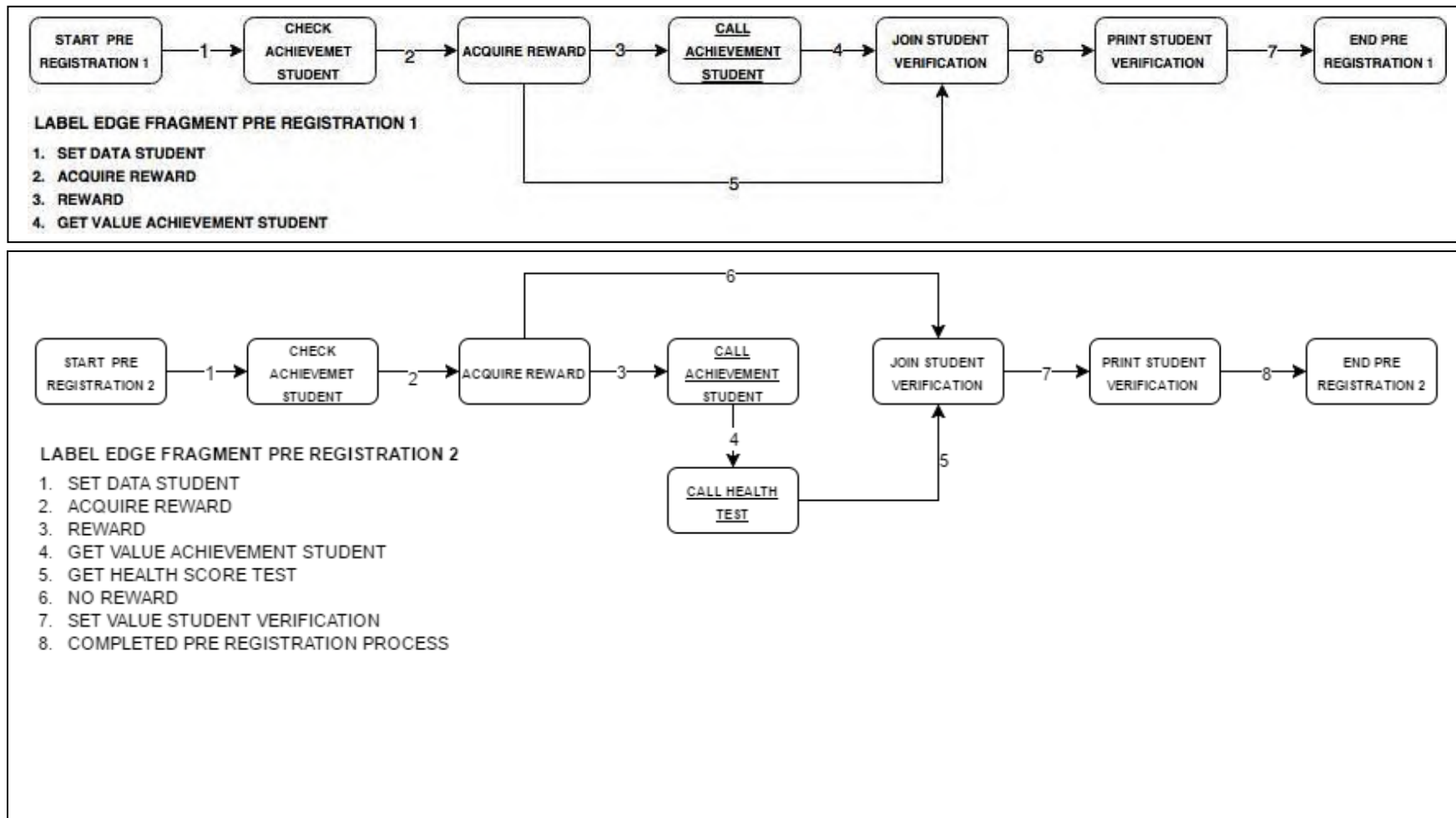
diuji memungkinkan untuk memeriksa kesamaan makna dengan nilai diantara 0–

1. Hasil penelitian ini menunjukan peningkatan nilai akurasi pada perhitungan WDAG *Similarity* sebesar 17,5% menggunakan Tekstual *Similarity* PLSA .

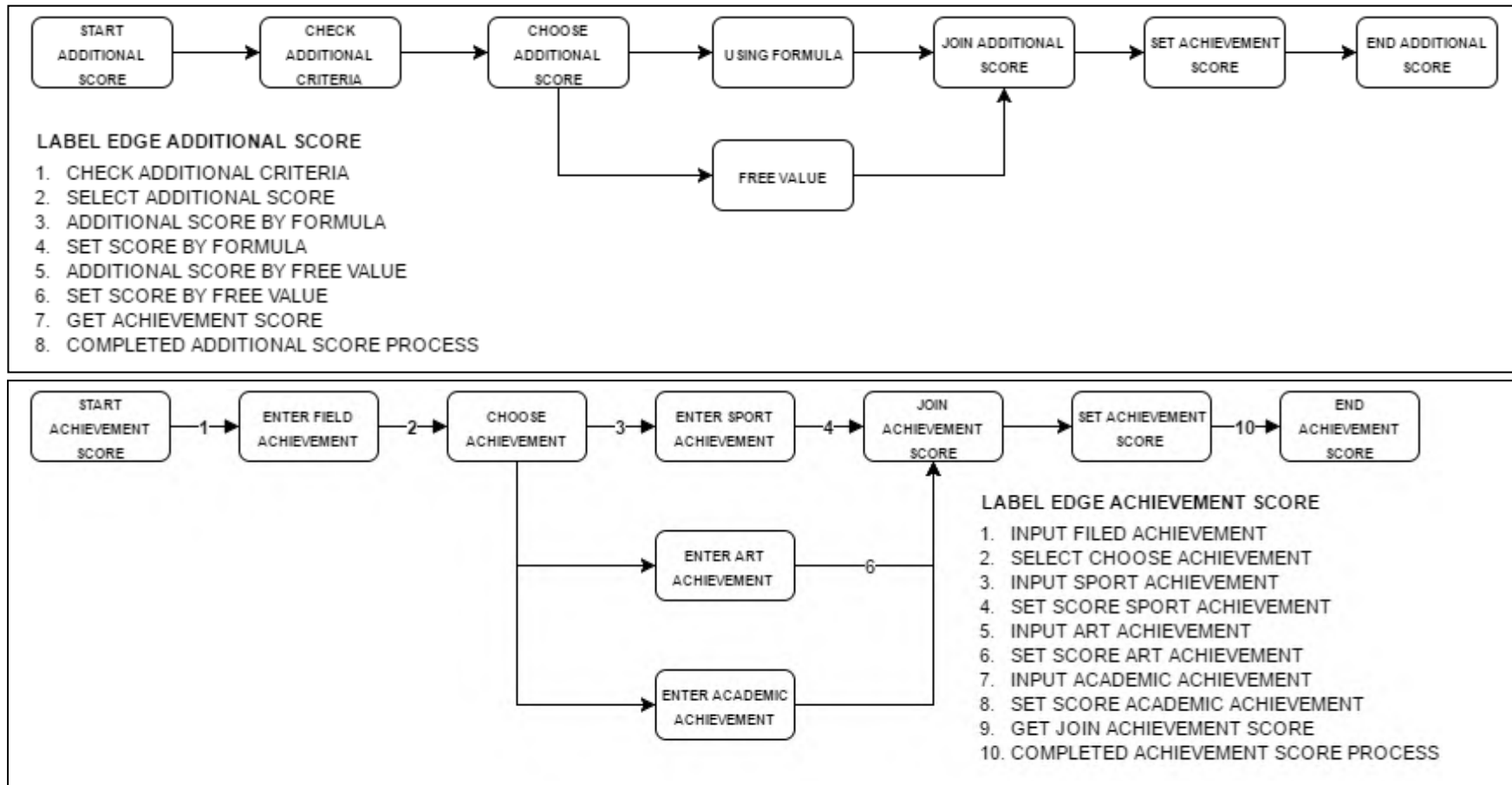
(halaman ini sengaja dikosongkan)

LAMPIRAN 1. Model PPDB

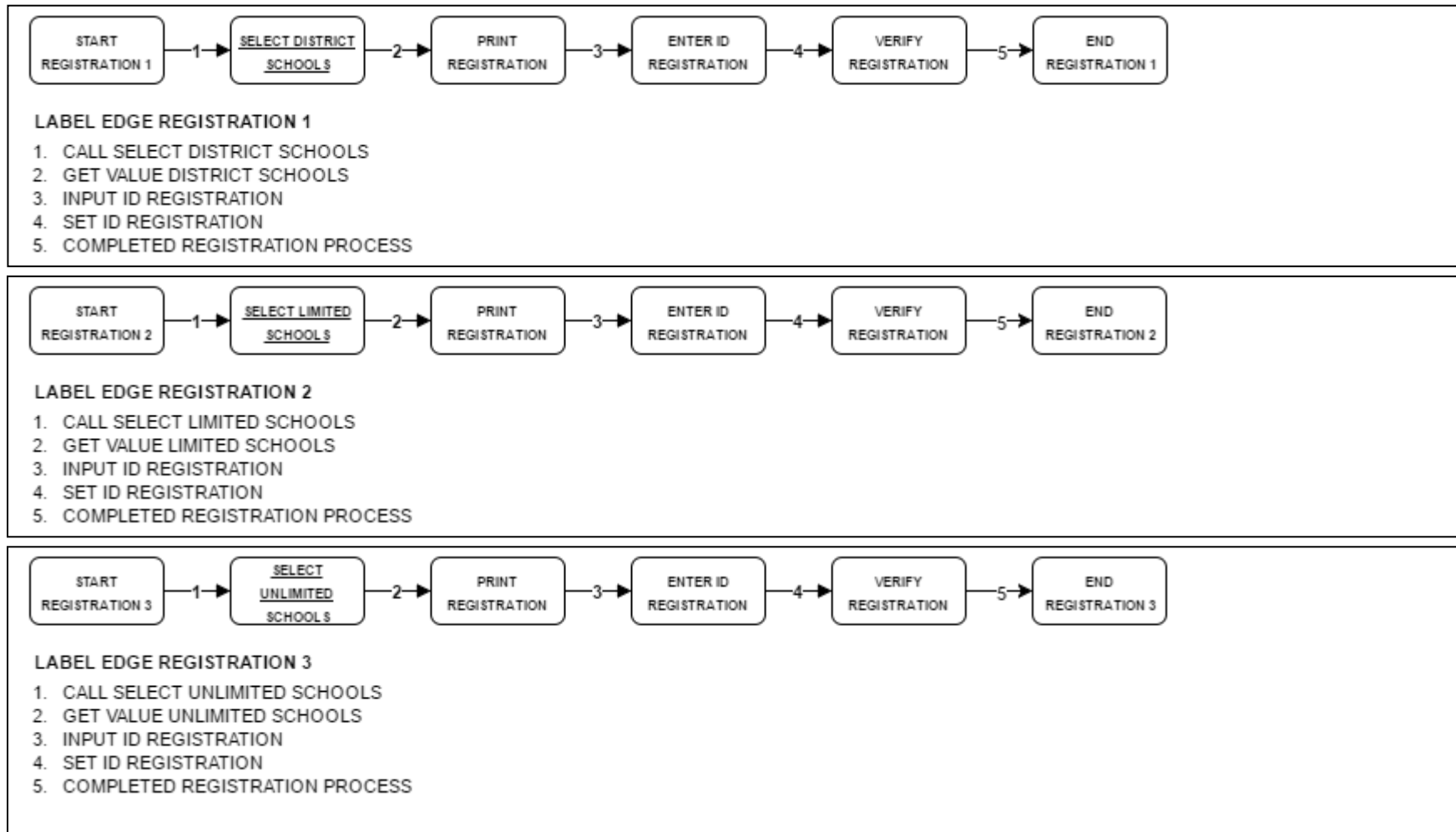
L1.1 Contoh Model Pre Registration



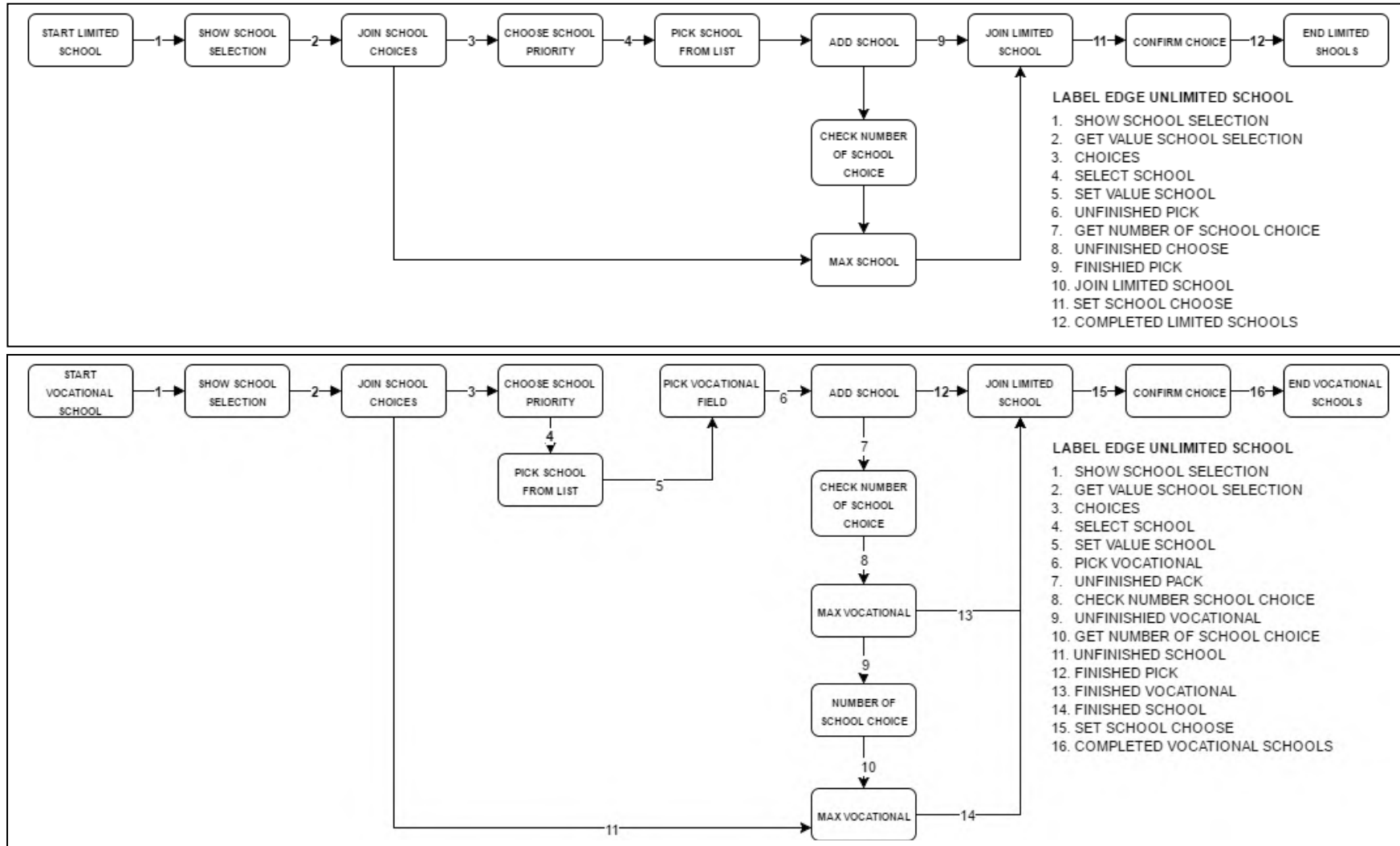
L1.2 Contoh Variasi Model Pre Registration



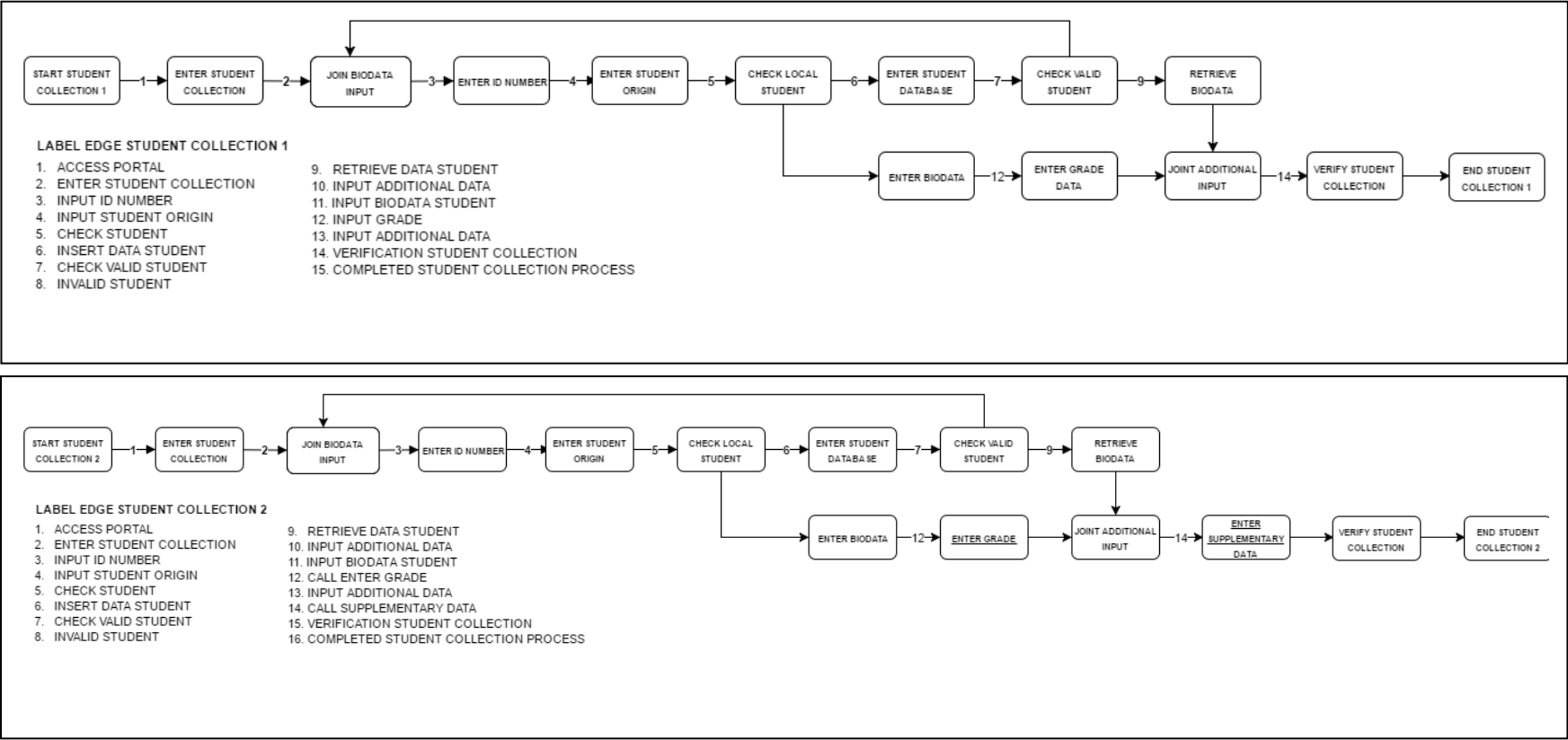
L1.3 Contoh Model Registration



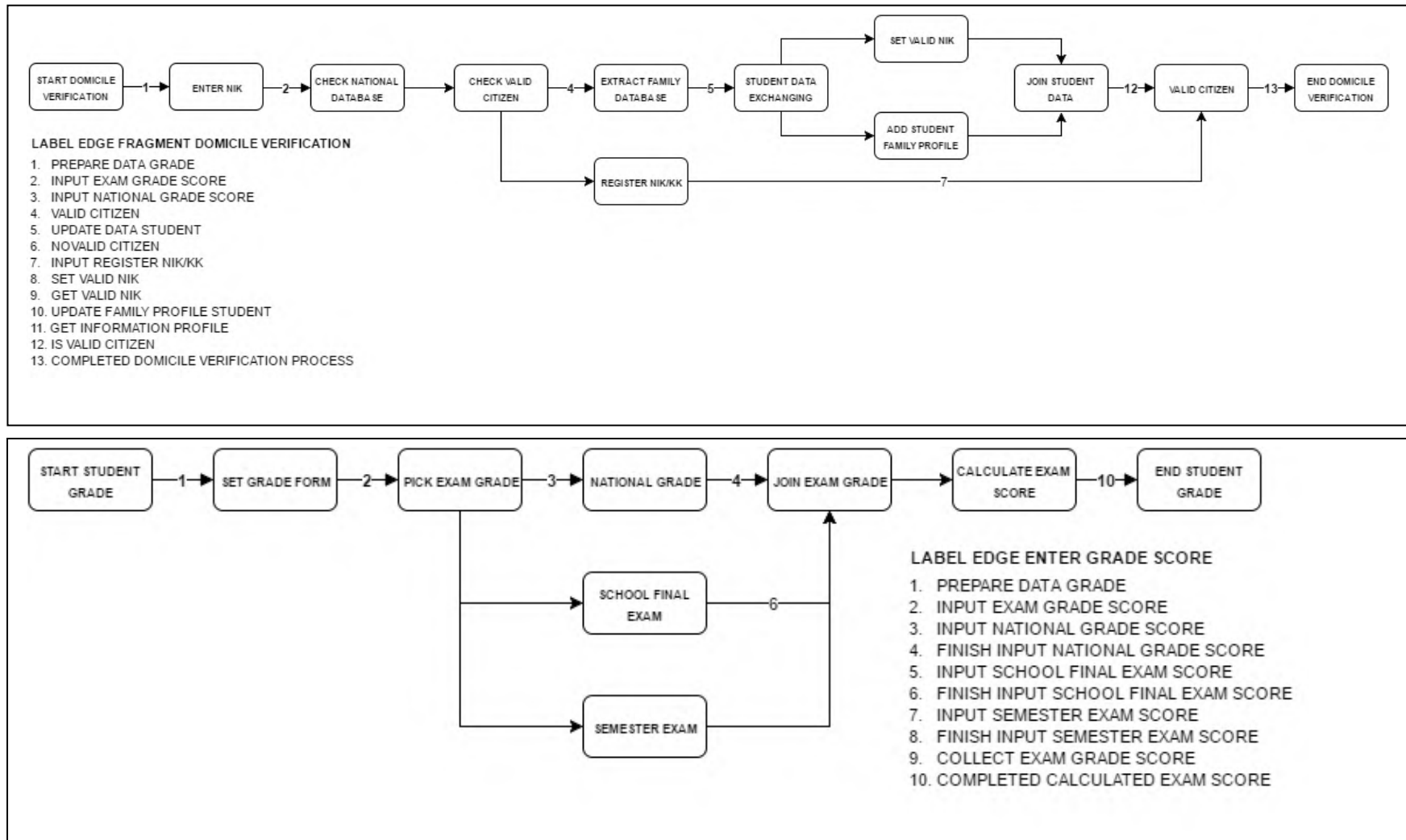
L1.4 Contoh Variasi Model Registration



L1.5 Contoh Model Student Collection



L1.6 Contoh Variasi Model Student Collection



BAB V

PENUTUP

Pada bab ini dijelaskan mengenai kesimpulan yang didapat setelah melakukan serangkaian uji coba serta saran untuk pengembangan penelitian lebih lanjut.

5.1 Kesimpulan

Pada sub-bab ini dipaparkan kesimpulan yang dapat diambil berdasarkan hasil percobaan dan analisa penelitian yang dilakukan terhadap metode yang diusulkan, kesimpulan tersebut adalah sebagai berikut:

1. Metode perhitungan similaritas gabungan menggunakan WDAG PLSA mampu meningkatkan akurasi dibandingkan menggunakan WDAG saja. Hasil pengujian menunjukan metode ini mampu meningkatkan akurasi sebesar 40%.
2. Metode *Tekstual Similarity* menggunakan *Probabilistic Latent Semantic Analysis* (PLSA) menggunakan data label *string* dari proses bisnis PPDB menghasilkan nilai lebih baik dibandingkan menggunakan metode *Vector Space Model* (VSM) sebesar 0,047% dan *Latent Semantic Analysis* (LSA), sebesar 0,095%.
3. Metode PLSA dapat memeriksa label *string* yang mempunyai makna yang berdekatan dengan menggunakan nilai probabilitas dari kategori/topik.
4. Metode *Process Vector* dapat digunakan untuk menentukan bobot dalam WDAG dengan menggunakan *Distance Weight* dan *Execution Probability*.

5.2 Saran

Berdasarkan hasil yang didapatkan dari penelitian ini, ada beberapa saran yang berguna untuk penelitian-penelitian mendatang, yaitu:

1. Perlu adanya eksplorasi metode analisa similaritas gabungan dengan menggunakan analisa struktural dan tekstual sebagai pembanding.
2. Perlunya dibangun kamus kata (*Bag of Word*) proses bisnis, sehingga dapat digunakan dalam pemodelan topik menggunakan PLSA, dengan

harapan dapat memetakan kata-kata yang memiliki kedekatan makna lebih baik dan dapat dijadikan acuan untuk penelitian selanjutnya.

3. Perlu adanya metode transformasi pembandingan untuk merepresentasikan model proses ke dalam representasi WDAG.
4. Perlu adanya metode pembobotan lainnya sebagai pembandingan metode *Process Vector*.
5. Pada penelitian berikutnya disarankan ada penambahan proses otomatisasi dalam mentransformasikan *Triconnected Abstraction* ke dalam WDAG.

DAFTAR PUSTAKA

- Aalst, W. M. P. Van Der. (1998). The Application of Petri Nets to Workflow Management. *Journal of Circuits, Systems, and Computers*, 8(01), 21–66.
- Bustami. (2014). Penerapan Algoritma Naïve Bayes Untuk Mengklasifikasi Data Nasabah Asuransi. *Jurnal Informatika*, 8(1), 884–898.
- Cardoso, J. (2008). Business process control-flow complexity: Metric, evaluation, and validation. *International Journal of Web Services Research*, 5(2), 49–76.
- Dijkman, R., Dumas, M., Dongen, B. Van, Reina, K., & Mendling, J. (2011). Similarity of business process models: Metrics and evaluation. *Information System*, 36(2), 498–516. <http://doi.org/10.1016/j.is.2010.09.006>
- Dijkman, R., Dumas, M., & García-Bañuelos, L. (2009). Graph matching algorithms for business process model similarity search. *International Conference on Business Process Management*, 48–63. http://doi.org/10.1007/978-3-642-03848-8_5
- Dumais, S. T. (1995). Latent semantic indexing (LSI): TREC-3 report. *Nist Special Publication Sp*, 219–230. Retrieved from [http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Latent+Semantic+Indexing+\(+LSI+\):+TREC-3+Report#0](http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Latent+Semantic+Indexing+(+LSI+):+TREC-3+Report#0)
- Hofmann, T. (1999). Probabilistic Latent Semantic Indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 50–57). New York: ACM.
- Jin, J. (2006). *Similarity of weighted directed acyclic graphs*. The University Of New Brunswick, 34–50.
- Jung, J. Y., Bae, J., & Liu, L. (2009). Hierarchical Clustering of Business Process Models. *International Journal of Innovative Computing, Information and Control*, 12(5), 4501–4511.
- Landauer, T. K., Folt, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, 25(2), 259–284. <http://doi.org/10.1080/01638539809545028>
- Ling, J., Zhang, L., & Feng, Q. (2014). An Improved Structure-based Approach to Measure Similarity of Business Process Models. *SEKE*, 377–380.
- Magerman, T., Van Looy, B., Baesens, B., & Debackere, K. (2011). *Assessment of Latent Semantic Analysis (LSA) text mining algorithms for large scale mapping of patent and scientific publication documents*. Leuven: K.U.Leuven - Faculty of Business and Economics, 1–77.
- Munoz-Gama, J., Carmona, J., & Aalst, W. M. P. Van Der. (2014). Single-Entry Single-Exit decomposed conformance checking. *Information Systems*, 46, 102–122. <http://doi.org/10.1016/j.is.2014.04.003>
- Osmanli, O. N. (2010). *A SINGULAR VALUE DECOMPOSITION APPROACH FOR RECOMMENDATION SYSTEMS*. The graduate school of naturel and applied sciences of middle east technical university. Middle East Technical University, 1–67.
- Polyvyanyy, A. (2012). *Structuring Process Models*. University of Postdam, Postdam, 35–59.
- Pramono, D., Setiawan, N. Y., Sarno, R., & Sidiq, M. (2013). Physical activity recommendation for diabetic patients based on ontology. *The 7th*

- International Conference on Information & Communication Technology and Systems*, 27–32. Retrieved from http://icts.if.its.ac.id/openaccess/2013/files/PP_6_PAPER_66.pdf
- Rosario, B. (2000). Latent Semantic Indexing: An overview. *Techn. Rep. INFOSYS*, 240, 1–16.
- Sarno, R., Djeni, C. A., Mukhlas, I., & Sunaryono, D. (2015). Developing a workflow management system for enterprise resource planning. *Journal of Theoretical and Applied Information Technology*, 72(3), 412–421. Retrieved from <http://www.scopus.com/inward/record.url?eid=2-s2.0-84923658674&partnerID=40&md5=5a38438f73110244470248466b0961f8>
- Vanhatalo, J., Völzer, H., & Koehler, J. (2009). The refined process structure tree. *Data {&} Knowledge Engineering*, 68(9), 793–818. <http://doi.org/10.1016/j.datak.2009.02.015>
- Weske, M. (2007). *Business Process Management Architectures. Business Process Management: Concepts, Languages, Architectures*. Potsdam, Germany: Springer, 305–343.

(halaman ini sengaja dikosongkan)

BIODATA PENULIS



INDRA GITA ANUGRAH lahir di Surabaya pada tanggal 22 Nopember 1986, Putra pertama dari dua bersaudara. Pendidikan Strata 1 di Institut Teknologi AdhiTama Surabaya Fakultas Teknologi Informasi Jurusan Informatika dan lulus tahun 2009.

Kemudian pada tahun 2014 dengan rahmat Allah SWT, penulis diterima di Magister Teknik Informatika Fakultas Teknologi Informasi ITS melalui jalur mandiri dan berhasil menyelesaikan studi pada tahun 2016 dengan bidang minat MI (Manajemen Informasi). Penulis memiliki ketertarikan dalam bidang pembuatan perangkat lunak, dan pemodelan, terutama dalam pemodelan proses bisnis. Penulis dapat dihubungi melalui email : indragitaanugrah@yahoo.com

(halaman ini sengaja dikosongkan)